

# AIM: Amending Inherent Interpretability via Self-Supervised Masking

Eyad Alshami<sup>1,2</sup> Shashank Agnihotri<sup>3</sup> Bernt Schiele<sup>1,2</sup> Margret Keuper<sup>1,3</sup>

<sup>1</sup>Max-Planck-Institute for Informatics, Saarland Informatics Campus, Germany

<sup>2</sup>RTG Neuroexplicit Models of Language, Vision, and Action, Saarbrücken, Germany

<sup>3</sup>Data and Web Science Group, University of Mannheim, Germany

{ealshami, schiele, keuper}@mpi-inf.mpg.de, shashank.agnihotri@uni-mannheim.de

## Abstract

*It has been observed that deep neural networks (DNNs) often use both genuine as well as spurious features. In this work, we propose ‘‘Amending Inherent Interpretability via Self-Supervised Masking’’ (AIM), a simple yet interestingly effective method that promotes the network’s utilization of genuine features over spurious alternatives without requiring additional annotations. In particular, AIM uses features at multiple encoding stages to guide a self-supervised, sample-specific feature-masking process. As a result, AIM enables the training of well-performing and inherently interpretable models that faithfully summarize the decision process. We validate AIM across a diverse range of challenging datasets that test both out-of-distribution generalization and fine-grained visual understanding. These include general-purpose classification benchmarks such as ImageNet100, HardImageNet, and ImageWoof, as well as fine-grained classification datasets such as Waterbirds, TravelingBirds, and CUB-200. AIM demonstrates significant dual benefits: interpretability improvements, as measured by the Energy Pointing Game (EPG) score, and accuracy gains over strong baselines. These consistent gains across domains and architectures provide compelling evidence that AIM promotes the use of genuine and meaningful features that directly contribute to improved generalization and human-aligned interpretability.*

## 1. Introduction

Modern deep neural networks (DNNs) have achieved remarkable success across domains such as Natural Language Processing and Computer Vision. Despite their impressive performance metrics, these models often use spurious features that happen to correlate with target labels in training data but lack causal relevance to the task. This phenomenon, sometimes called ‘Clever Hans’ behavior [20, 35]. A classic example is classification models trained to distinguish between ‘land birds’ and ‘wa-

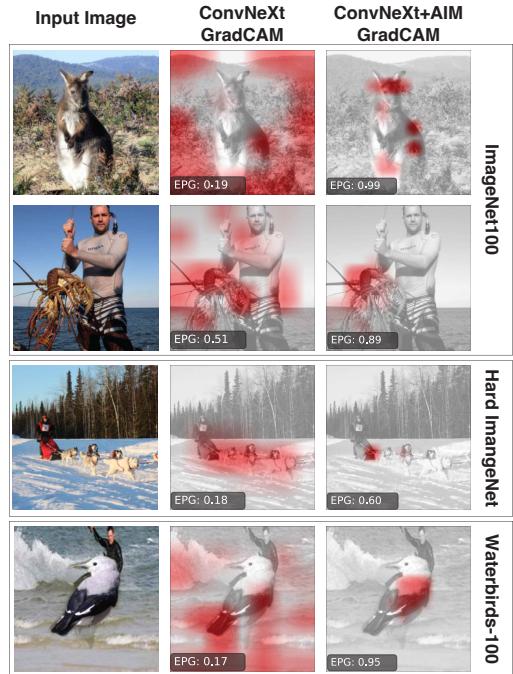


Figure 1. AIM uses self-supervised masking to focus more on the object of interest, relying only on the image label. As shown, and in terms of attribution localization, it outperforms baseline methods, even in challenging scenarios like the WaterBirds dataset.

ter birds’ for the WaterBirds dataset [28, 33] that often learn to classify the background environment rather than the birds themselves, resulting in poor generalization when birds appear in atypical habitats. It is therefore desired that models leverage *genuine* features, that are distinctive, class specific, and are localized on the object. Recent research has shown that, while DNNs exhibit dependence on spurious features, they simultaneously acquire some genuine features [18]. This key insight suggests an opportunity: How can we promote the models’ utilization of genuine features while suppressing spurious ones? Prior works

proposed using extra annotations in the form of bounding boxes, segmentation masks, or other guiding mechanisms [9, 11, 12, 22, 26, 28, 30, 34, 38, 43, 46, 53]. These mechanisms help focus the model on genuine features while ignoring spurious ones during training. However, getting these extra annotations is often nonviable.

Thus, we propose AIM (Amending Inherent Interpretability via Self-Supervised Masking), a method that encourages the model to focus on genuine features and ignore spurious ones without needing annotations beyond image labels. AIM employs a self-supervised masking mechanism that systematically identifies and prioritizes dependable feature maps of convolutional neural networks by masking out spurious features and retaining only dependable ones. Unlike previous approaches that rely on external attribution methods or require expensive additional annotations, AIM operates by applying learnable binary masks to feature maps, allowing the model itself to determine which regions to retain or discard based on task performance. Our conjecture is that, when forced to select a subset of spatial features prior to making a classification decision, a model will consider those features most dependable that generalize best, i.e., that are genuine. We confirm this hypothesis using various analyses, including Energy Pointing Game (EPG) scores and evaluations using challenging datasets that provide many spurious cues.

The AIM mechanism involves both a bottom-up processing of visual information through convolutional layers and a top-down pathway that refines feature selection. This feature refinement progressively identifies and filters out spurious features while preserving dependable ones. Importantly, this masking mechanism makes the model’s decision process transparent: what is visible in the final feature representations directly causes the classification outcome, creating inherently interpretable models rather than relying on post-hoc interpretability methods.

We evaluate AIM on challenging datasets specifically designed to test models’ resilience to spurious features, including Waterbirds and TravelingBirds [19]. These datasets present scenarios where background features strongly correlate with class labels during training but not during testing, challenging the models’ out-of-distribution (OOD) generalization capabilities. We also validate our approach on standard fine-grained classification benchmarks such as CUB-200 [50]. Across these evaluations, AIM demonstrates significant improvements in localization accuracy (measured by the Energy Pointing Game score). It also improves classification performance in OOD scenarios, showcasing improved generalization through the use of genuine features. Our results demonstrate that by encouraging the model to narrow its selection of spatial features for classification, it improves its focus on dependable ones. AIM achieves this through a masking mechanism that produces

inherently interpretable models without compromising task performance. The spatial masks learned by AIM provide clear visual evidence of the features driving the model’s decisions, establishing a “what you see causes what you get” relationship between the used features and predictions.

The primary contributions of this work are threefold:

- We propose a simple yet effective self-supervised masking mechanism that guides DNNs to utilize dependable features over spurious alternatives, yielding inherently interpretable decisions while requiring only image labels.
- We demonstrate that our approach significantly improves the models’ ability to localize genuine features, as quantified by Energy Pointing Game scores across multiple datasets.
- We show through extensive experiments that AIM yields improvements in challenging out-of-distribution generalization scenarios where spurious features typically cause models to fail.

## 2. Related Work

The prevalence of spurious correlations in DNNs, coupled with their increasing deployment in critical applications, has prompted extensive research in interpretability.

**Model Guidance and Attribution Methods.** Attribution methods generate attention maps [3, 4, 17, 29, 36, 39, 41, 51] that highlight important input regions contributing to the final decision, aiding in the identification of erroneous reasoning by the model. Model guidance builds on these methods to align vision systems with ground truth guidance sources [11, 12, 31, 38, 44, 45], ensuring models are ‘right for the right reasons’ [31]. This strategy relies on extra annotations, such as bounding boxes or attention maps [9, 11, 12, 22, 28, 30, 43, 53], which can be expensive and imperfect [11]. Several methods aim to reduce the dependency on extra annotations. For instance, cost-effective model guidance can be achieved using only a small fraction of annotated images [30]. When no extra annotations are available, approaches like [31] iteratively generate models with different reasoning but still require expert selection. Others improve explanations annotation-free simply by tuning the classification head’s loss function (e.g., using binary cross-entropy) [10]. Alternatively, [2] fine-tunes the model by masking discriminative features identified by the trained model. Recent work [18] observed that DNNs, while relying on spurious correlations, still learn genuine features. However, their method assumes prior knowledge of the spurious correlation.

**Content-Based Conditional Operations.** Methods in this domain [14, 42, 49] constrain the model to prioritize relevant spatial regions within the input features without requiring additional annotations. Some apply masking in the feature maps during the forward pass [14, 49], pushing learned

features to focus on ‘regions of interest’. Others apply masking in the input image domain [42]. For example, [14] uses mask estimators with the Gumbel-softmax trick to predict binary masks that identify crucial areas and preserve them in higher resolutions. Similarly, [49] employs mask estimators with the Gumbel-softmax trick to select and process only important spatial regions, accelerating inference. Both works [14, 49] achieve spatial selection by progressively applying the masking strategy as the input moves through the network, from the initial layers all the way to the final layers, in a bottom-up fashion. A common problem reported by both works is that the generated masks tend to be fully active, requiring additional loss functions to push them to be sparse. We hypothesize that the issue of fully active masks stems from the bottom-up approach itself. In contrast, AIM allows the network to reassess the generated feature maps across the entire architecture, utilizing the top-down approach. This naturally produces sparse masks and enables the model to use spatially sparse feature maps, enhancing explainability.

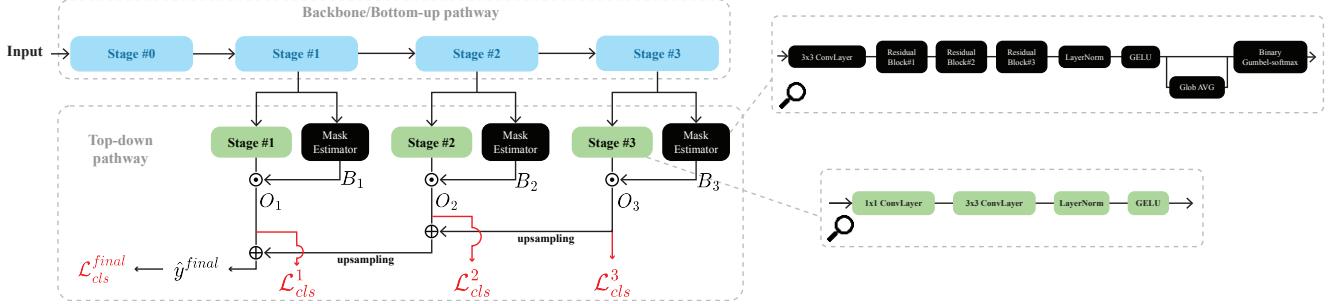
### 3. Proposed AIM Method

Our method comprises two main pathways: a convolutional neural network (CNN) for the bottom-up pathway with multiple encoding stages, and a top-down pathway with corresponding self-supervised masking modules. Our work proposes a novel top-down pathway that helps the CNN focus on genuine features and ignore the spurious ones learned by the encoding stages of the bottom-up pathway. It achieves this using two main components: first, a mask estimator that sparsifies the feature maps from the encoding stages, and second, a pathway that combines the sparse feature maps from different encoding stages. The following describes these main components of our method in detail. Please note, we address the task of image classification involving  $C$  classes, given a dataset  $\{(x_i, y_i)\}_{i=1}^n$  of size  $n$ , where  $x_i \in \mathbb{R}^{h \times w \times 3}$  represents input images and  $y_i$  their corresponding labels. Our approach does not require any additional annotations.

**Overall Architecture.** Our architecture builds on the Feature Pyramid Network (FPN) framework [21], adopting its top-down pathway structure. As illustrated in Figure 2, the model operates through two distinct pathways: a bottom-up pathway for hierarchical feature extraction and a top-down pathway for multi-scale feature integration. The bottom-up pathway employs a backbone network to generate hierarchical feature representations from input images. The top-down pathway iteratively combines these multi-scale features, propagating semantic information from the final high-level feature maps backward to earlier stages. Unlike the original FPN, which extends the top-down pathway to the highest-resolution initial feature map, we introduce

a hyperparameter to control the termination depth of this pathway. This modification enables systematic analysis of how varying degrees of semantic detail from intermediate layers affect both the guidance mechanism and overall network performance. For example in our baseline ConvNeXt-Tiny backbone [24], comprising four convolutional stages  $\{S_0, S_1, S_2, S_3\}$  (where  $S_3$  marks the final stage) with spatial resolutions  $\{56^2, 28^2, 14^2, 7^2\}$ , the top-down pathway stages  $\{T_0, T_1, T_2, T_3\}$  mirror these resolutions. We study how integrating intermediate top-down features (*e.g.*  $T_1, T_2$ ) to the final stage  $T_3$  enhances the model’s self-guiding capability. In the top-down pathway, shown in Figure 2, each stage receives the output of the corresponding stage in the bottom-up pathway, processes it, and prepares it for integration with outputs from the subsequent stage in the top-down sequence. This involves passing the feature maps through: 1) a  $1 \times 1$  convolutional layer to align channel dimensions across stages, 2) a  $3 \times 3$  convolutional layer for spatial refinement and 4) Layer Normalization and GELU activation. The computational overhead and the increase in the number of parameters are moderate, as summarized in Appendix A.2 in the appendix.

**Mask Estimation.** To enable spatial selection of dependable feature regions, we incorporate a learnable mask estimator at every stage of the top-down pathway. Each estimator consists of a lightweight convolutional neural network (CNN) that predicts a binary mask using the Gumbel-Softmax trick [49]. While prior work by Verelst and Tuytelaars [49] and Hesse et al. [14] employs a bottom-up masking strategy (where binary masks iteratively select spatial regions at each stage) we adapt this concept to our top-down framework. Our empirical results demonstrate that this adaptation inherently produces spatially sparse and focused masks, enabling the network to prioritize salient regions without an additional supervision signal. As shown in Figure 2, the architecture of the mask estimators used in our method begins with a  $3 \times 3$  convolutional layer, followed by three residual blocks that each utilize  $3 \times 3$  convolutional layers. The output is then split into two branches: an identity branch and a global average pooling operation to capture global context information. The global context vector is expanded and concatenated with the output of the identity branch. Finally, a  $1 \times 1$  convolutional layer is applied to generate the final single-channel feature maps. This single layer is passed through the Gumbel-softmax module adapted from [49] to generate the final binary mask. Each mask estimator uses the output from the corresponding stage of the backbone network as its input and generates a binary mask  $B$  with the same spatial resolution as the input feature maps. These masks highlight model-regarded dependable features in the feature maps generated by the corresponding convolutional stage. Formally, at each stage  $\ell$  in the architecture, the bottom-up stage  $S_\ell$  processes its



**Figure 2. Abstract Diagram of the [backbone]+AIM Architecture.** The architecture consists of a **bottom-up** backbone and a **top-down** masking pathway. The bottom-up pathway has four encoding stages ( $L = 0$  to  $L = 3$ ). The top-down pathway mirrors this structure, with each stage  $T$  corresponding to a bottom-up stage  $L$ . Each top-down stage has two parallel branches: one estimates a binary mask via a convolutional network with Gumbel-softmax, and the other processes features using a structure inspired by [21]. The estimated binary mask is element-wise multiplied with the processed features to create a spatially sparse feature map. These sparse maps are then iteratively combined with the output of the subsequent top-down stage through element-wise summation.

input  $x_\ell$ , producing feature maps  $S_\ell(x_\ell)$ . These feature maps are then passed through the two branches at the corresponding top-down stage. The first branch, denoted as  $T_\ell$ , is responsible for unifying the number of channels and post-processing the feature maps to prepare them for merging. It takes  $S_\ell(x_\ell)$  as input and produces the transformed feature maps  $T_\ell(S_\ell(x_\ell))$ . The second branch is responsible for generating the binary mask, which highlights dependable features identified by the model. This process involves two steps. First, a soft attention decision map  $A_\ell \in \mathbb{R}^{w_\ell \times h_\ell}$  is computed by a simple mask estimating module  $M$ :

$$A_\ell = M(S_\ell(x_\ell)) \quad (1)$$

Next, and following [14], to obtain the binary mask, a binary Gumbel-softmax module  $G$  is applied element-wise to  $A_\ell$ , resulting in  $B_\ell \in \{0, 1\}^{w_\ell \times h_\ell}$ :

$$B_\ell = G(A_\ell) \quad (2)$$

Finally, the spatially sparse output of the top-down stage  $\ell$  is then computed by element-wise multiplying the processed feature maps from the feature processing branch with the binary mask from the mask estimator:

$$O_\ell = T_\ell(S_\ell(x_\ell)) \odot B_\ell \quad (3)$$

where  $\odot$  denotes element-wise multiplication. This operation results in  $O_\ell$ , which retains only the dependable features as determined by the binary mask, effectively filtering out spatial regions with spurious features.

**Top-Down Sparse Feature Fusion** Along the top-down pathway, the output of each stage is up-sampled through nearest-neighbor interpolation to match the resolution of the next lower stage and then merged with its feature maps through element-wise summation. This continues down the pathway, progressively integrating multi-scale information. The final aggregated feature maps, containing reliable multi-scale features, are used for classification.

**Supervising The Mask Estimators.** Since our method does not rely on additional annotations, we supervise the mask estimators indirectly using the classification loss computed on the masked feature maps  $O_\ell$ . At each stage of the top-down pathway, this loss is applied before merging with higher-stage feature maps. This strategy ensures that each stage independently identifies and learns important regions based solely on the feature maps available up to that stage.

We pass these feature maps through a classifier  $f_\ell$  to obtain the predicted class probabilities  $\hat{y}^{(\ell)}$ :

$$\hat{y}^{(\ell)} = f_\ell(O_\ell) \quad (4)$$

And then compute the classification loss at each stage,  $\mathcal{L}_{cls}^{(\ell)}$ . At the final stage, the merged sparse feature maps, combining outputs from all previous stages, are passed through the final classifier  $f_{final}$  to obtain the final predicted class probabilities  $\hat{y}^{final}$ :

$$\hat{y}^{final} = f_{final}(O_{final}) \quad (5)$$

During training, our model self-guides to highlight spatial regions with dependable features within each stage’s output. By enabling the network to select these regions across all layers, we empirically show that this improves classification performance and enables the reliance on spatially sparse maps, leading to transparent and inherently interpretable decision-making.

**Optional Mask Annealing.** Our approach naturally generates sparse masks, but enforcing additional sparsity during training on challenging out-of-distribution datasets, such as Waterbirds and TravelingBirds, improved performance and produced more focused masks. This is done by applying a mean-squared loss on the number of active elements in the generated masks using a threshold  $\tau_i$  as follows:

$$\mathcal{L}_{masks_i} = (r_i - \tau_i)^2 \quad \text{where} \quad r_i = \frac{\sum_{j=0,k=0}^{B_h^i, B_w^i} \mathbb{1}(B_{j,k}^i = 1)}{\sum_{j,k}^{B_h^i, B_w^i} 1} \quad (6)$$

Where  $r_i$  is the ratio of active elements in the generated binary mask  $B^i$ , and  $\tau_i$  is a threshold hyperparameter that can be selected for each stage’s mask estimator. We use a masking annealing technique to help the network gradually adapt to sparsity constraints without disrupting learning. Training begins with fully active masks (i.e.  $\tau_i = 1.0$ ) and progressively lowers the active-area loss threshold each epoch until it reaches a target value (e.g.  $\tau_i = 0.35$ ), which is then held for the remainder of training. The annealing duration is treated as a hyperparameter. This strategy improves mask quality and stabilizes learning. (For more details, see Appendix D). Based on this setup, the final loss is defined as:

$$\mathcal{L}_{Total} = \lambda \sum_{i=L}^{\ell} \mathcal{L}_{masks_i} + \sum_{i=L}^{\ell} \mathcal{L}_{cls}^{(i)} \quad (7)$$

Here,  $L$  is the index of the highest stage, and  $\ell$  denotes the final stage we aim to reach in the top-down pathway. The parameter  $\lambda$ , set to 6 in all experiments, weights the mask’s active-area loss to be on the same order of magnitude as the classification loss. For a full list of hyperparameters, see Appendix A.

## 4. Experiments

In this section we show that AIM helps to retain in-domain performance while significantly boosting out-of-domain performance. First, however, we describe some important implementation details (more details in Appendix A).

**AIM Architectural Variants.** As detailed in Section 3, we parameterize the top-down pathway’s depth by the number of stages traversed, denoting these variants as “Backbone+AIM [index]”, where *index* signifies the stage  $T$  where propagation ceases. For instance, with ResNet50, which has five convolutional stages, including the stem cell, we implement two main variants: ResNet50+AIM (2) incorporates feature maps from stages 4, 3, and 2, while ResNet50+AIM (3) includes only feature maps from stages 4 and 3.

**Baselines.** To evaluate our approach, we tested the effect of applying AIM across various backbone architectures by comparing the performance of each backbone with and without AIM integration. We utilized ConvNeXt-tiny [24], ResNet-50 [13], and ResNet-101 [13]. All of these models are pre-trained on ImageNet-1k [5].

**Evaluation Metrics.** To assess how effectively AIM promotes dependable feature learning, we evaluate spatial localization using the Energy Pointing Game (EPG) score [51]. This metric, based on attribution maps (e.g., GradCAM [36], Guided GradCAM [37], or other attribution methods) and ground-truth binary masks, calculates the ratio of attribution within the mask’s active region to the total attribution. For implementation details, see Ap-

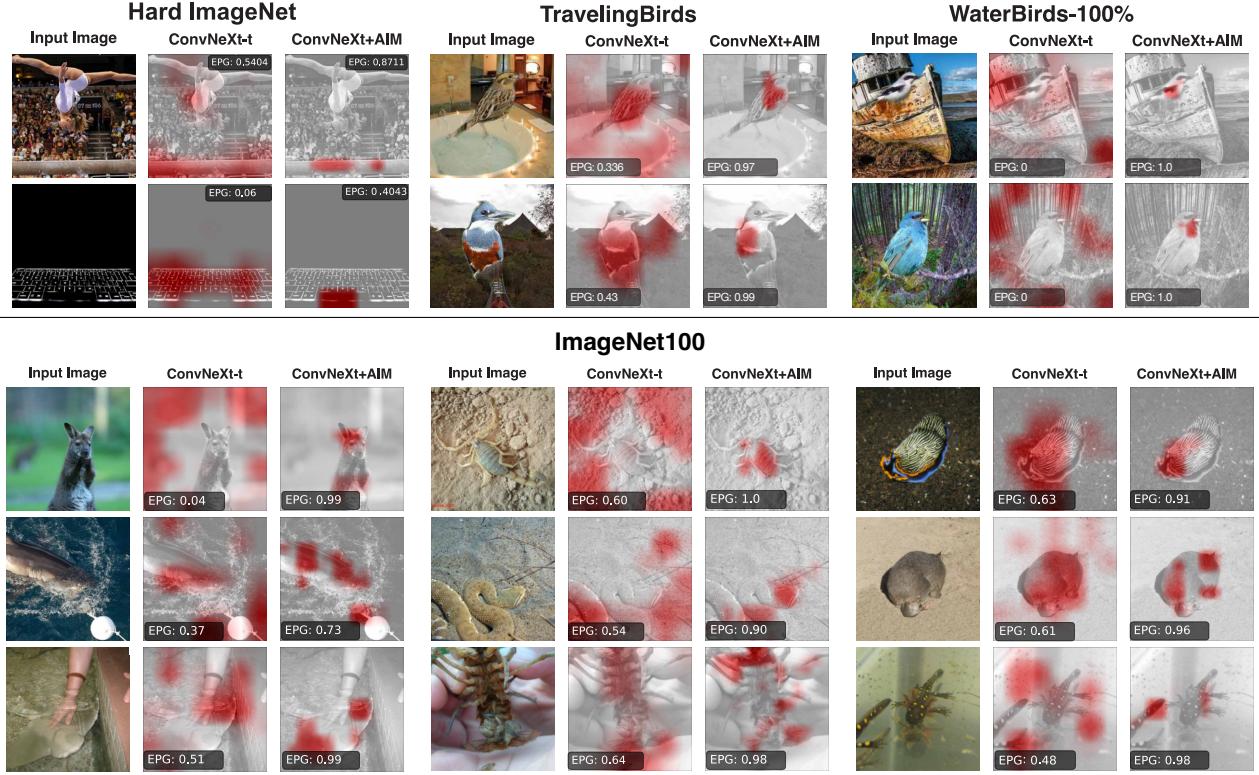
pendix A.5. In addition, we show that our method not only preserves but also improves classification accuracy.

**Mask Annealing via Active-Area Loss.** As detailed in Section 3, in addition to the stage  $T$  parameterization, we employ progressive mask sparsification via threshold annealing during training, reducing the initial 100% active area to either 35% or 25%. This introduces a new parameter,  $\tau$ , representing the final mask retention. Consequently, our models are denoted as “backbone+AIM [stage  $T$ , threshold  $\tau$ ]”. For example, ResNet50+AIM [2, 25%] signifies operation through stage  $T = 2$  with 25% mask retention.

**Datasets.** We evaluate our method on two categories of datasets. First, to test its robustness to spurious correlations, we use the synthetic Waterbirds (95% and 100% versions) [28, 33] and Travelingbirds [19] datasets. Second, to assess the broader adaptability and effectiveness of AIM, we use standard benchmarks including ImageNet100 [47], Hard-ImageNet [27], and the fine-grained Caltech-UCSD Birds-200-2011 (CUB-200-2011) [50]. Further details are provided in Appendix A.4.

## 4.1. Results on Out-Of-Domain Datasets

The Waterbirds and TravelingBirds datasets contain synthetic spurious correlations, causing models to incorrectly rely on background cues rather than foreground objects. As Figure 4 illustrates, our proposed AIM mechanism consistently surpasses baseline backbone models. Vanilla backbone performance significantly degrades due to these biases; however, models integrated with AIM show notable improvements in both EPG and accuracy across all tested out-of-domain datasets. The primary motivation of AIM, detailed in Section 1, is to enhance the localization of genuine image features, thereby improving interpretability without compromising accuracy. The improved accuracy across various backbones emerges as an additional beneficial outcome. The self-supervised masking strategy employed by AIM enables models to consistently identify and rely on dependable features. Figure 3 visually demonstrates this, contrasting baseline models, which are often distracted by misleading background cues, with AIM-equipped models that reliably emphasize dependable regions. To confirm this visual improvement is perceived by humans, we conducted a user study that showed participants preferred our model’s attribution maps over the baseline in 70.7% of cases ( $p < 0.00001$ ), providing strong evidence of more human-aligned interpretability (see Appendix D.6 for details). Quantitatively, higher EPG scores, computed using dataset-provided binary masks, confirm this improved localization. We also evaluate other attribution methods and observe similar EPG improvements on the Waterbirds-95% dataset using Guided GradCAM and Guided Backprop, as shown in Appendix B.2, further confirming the robustness of our localization gains. These scores validate our hy-

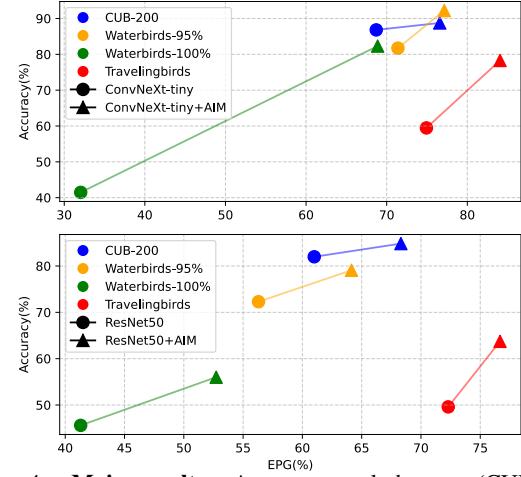


**Figure 3. Models amended with AIM consistently exhibit enhanced localization of genuine features, effectively suppressing spurious cues in both in-domain and out-of-domain scenarios.** A qualitative visualization of Grad-CAM heatmaps comparing baseline ConvNeXt-tiny models and ConvNeXt-tiny+AIM models across HardImagenet (classes shown: Balance Beam, Space Bar), TravelingBirds, WaterBirds-100%, and ImageNet100 datasets. The EPG scores, with a range of 0.0 to 1.0, are indicated on each heatmap. For more qualitative results, see Appendix C.

pothesis from Sec.1 regarding the dependability of features identified by AIM. As summarized visually in Figure 4 and detailed with precise metrics in Table 1, we see substantial EPG score improvements: approximately 6% for Waterbirds-95%, 30% for Waterbirds-100%, and 10% for TravelingBirds. Accuracy gains are equally notable, reaching around 10% for Waterbirds-95%, 40% for Waterbirds-100%, and 18% for TravelingBirds. Furthermore, Figure 5 provides a detailed per-sample analysis of EPG scores for baseline models versus models with AIM. It demonstrates that across all datasets, the majority of individual samples exhibit improved EPG scores. While overall improvements are substantial, a subset of examples maintained comparable performance, particularly when baseline EPG was already high, and a minimal number of instances showed slight EPG decreases. Additional comparisons against other relevant methods are provided in the Appendix B.4.

#### 4.2. Comprehensive Evaluation on Diverse Classification Tasks

To evaluate the adaptability and effectiveness of our proposed AIM mechanism, we conducted experiments on a range of classification benchmarks, from fine-grained tasks to broader, general-purpose datasets.



**Figure 4. Main results:** Across several datasets (CUB-200, Waterbirds-95%, Waterbirds-100%, Travelingbirds) and architectures (ConvNeXt-tiny, ResNet50) our AIM approach outperforms the respective baseline in both accuracy as well as interpretability as measured by the Energy Pointing Game score. On WaterBirds, following [33], we report the worst-group accuracy.

First, we tested our method on the CUB-200 dataset, which poses a challenging fine-grained classification task

**Table 1. Average Test Accuracies for ConvNeXt+AIM Configurations.** Comparison of the ConvNeXt-tiny baseline against our AIM-enhanced models on multiple benchmarks. The method shows significant gains, especially in worst-group accuracy on Waterbirds, highlighting its effectiveness in mitigating spurious correlations. All values are mean accuracy (%)  $\pm$  standard deviation.

Model	ImageNet100		Hard-ImageNet		Waterbirds				TravelingBirds	
	Acc	EPG	Acc	EPG	100%		95%		Acc	EPG
					WG-Acc	EPG	WG-Acc	EPG		
ConvNeXt-t	89.2 ( $\pm 0.1$ )	91.4 ( $\pm 0.3$ )	96.2 ( $\pm 0.2$ )	36.6 ( $\pm 0.5$ )	39.6 ( $\pm 5.4$ )	57.2 ( $\pm 6.0$ )	81.6 ( $\pm 3.2$ )	68.3 ( $\pm 3.2$ )	59.5 ( $\pm 0.8$ )	74.4 ( $\pm 0.6$ )
<b>ConvNeXt-t+AIM[1, 25%]</b>	90.5 ( $\pm 2.1$ )	91.5 ( $\pm 0.1$ )	<u>97.1 (<math>\pm 0.3</math>)</u>	<u>38.8 (<math>\pm 0.9</math>)</u>	73.6 ( $\pm 4.5$ )	<u>60.1 (<math>\pm 1.3</math>)</u>	91.2 ( $\pm 0.8$ )	<b>77.1 (<math>\pm 5.2</math>)</b>	77.1 ( $\pm 0.3$ )	79.0 ( $\pm 0.7$ )
<b>ConvNeXt-t+AIM[1, 35%]</b>	<u>90.5 (<math>\pm 0.1</math>)</u>	89.1 ( $\pm 0.2$ )	<b>97.3 (<math>\pm 0.2</math>)</b>	33.2 ( $\pm 0.5$ )	<u>77.1 (<math>\pm 4.4</math>)</u>	57.2 ( $\pm 1.3$ )	90.7 ( $\pm 0.7$ )	63.0 ( $\pm 1.2$ )	71.5 ( $\pm 1.3$ )	72.6 ( $\pm 1.5$ )
<b>ConvNeXt-t+AIM[2, 25%]</b>	90.1 ( $\pm 2.3$ )	<b>92.8 (<math>\pm 0.8</math>)</b>	96.8 ( $\pm 0.5$ )	<b>40.1 (<math>\pm 1.5</math>)</b>	74.0 ( $\pm 5.0$ )	58.0 ( $\pm 1.3$ )	<b>92.7 (<math>\pm 1.2</math>)</b>	<u>75.0 (<math>\pm 6.0</math>)</u>	<b>77.4 (<math>\pm 0.2</math>)</b>	<b>85.0 (<math>\pm 2.0</math>)</b>
<b>ConvNeXt-t+AIM[2, 35%]</b>	<b>90.7 (<math>\pm 0.0</math>)</b>	91.8 ( $\pm 0.1$ )	97.1 ( $\pm 0.1$ )	33.6 ( $\pm 1.1$ )	<b>78.1 (<math>\pm 2.3</math>)</b>	<b>68.5 (<math>\pm 3.6</math>)</b>	92.3 ( $\pm 0.6$ )	71.7 ( $\pm 6.4$ )	71.0 ( $\pm 0.4$ )	77.7 ( $\pm 0.4$ )

requiring models to identify subtle and localized visual features. As shown in Figure 4, incorporating our AIM mechanism improves localization performance significantly: ConvNeXt-tiny+AIM achieves an approximate 6% increase in EPG score over the baseline ConvNeXt-tiny model, while ResNet+AIM improves localization by around 9% compared to the baseline ResNet model. These localization improvements are accompanied by a slight accuracy increase of about 2–3%. For full details see Appendix B.3.

Next, to validate the broader applicability beyond the domain of bird datasets, we evaluated AIM on general-purpose image classification benchmarks. We conducted experiments using ConvNeXt-tiny on ImageNet100 [47] and the challenging HardImageNet [27]. As summarized in Table 1, our method shows consistent benefits across diverse domains. On ImageNet 100, it improves the EPG score by nearly 3 points while maintaining baseline accuracy, reinforcing our core claim of enhanced localization. On the more difficult HardImageNet, it boosts both

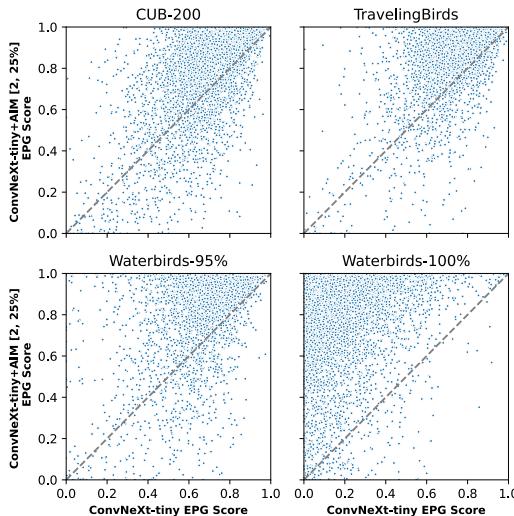


Figure 5. EPG scores per sample are plotted for baseline model (x-axis) v/s model amended with AIM (y-axis). We observe at a per-sample level for each of the four datasets that majority for the samples the EPG scores are improved by amending the model with our proposed AIM.

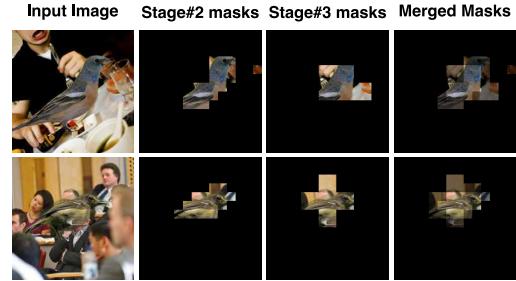


Figure 6. Illustration of masks learned at the two stages within the network (using ConvNeXt-tiny+AIM [2, 25%]), along with the final merged mask for each image. These merged masks highlight the sparse regions within the corresponding feature maps.

accuracy and EPG, confirming its robustness as a domain-agnostic mechanism.

## 5. Inherent Interpretability With Self-Supervised Masking

Our AIM mechanism offers inherent interpretability, which we visualize by depicting input images alongside their corresponding masks from the top-down pathway in Figure 6. This interpretability arises from the self-supervised masking performed by the mask estimator within AIM. Furthermore, combined masks clearly illustrate how sparse feature maps from different stages of the top-down pathway are merged.

Figure 7 visualizes the evolution of these masks over epochs. The mask estimator initially starts with random values, but as training progresses, relying solely on the classification loss from image labels, it learns to focus on dependable features within the feature maps. As discussed in Section 4, these dependable features correspond to genuine features, indicated by EPG scores. Conversely, if the model learns incorrect masking, low EPG scores reflect non-genuine features and result in lower accuracy scores. Since AIM’s self-supervised masking mechanism is part of the model’s forward pass, visualizing these masks directly reveals the basis of the model’s decisions. This establishes

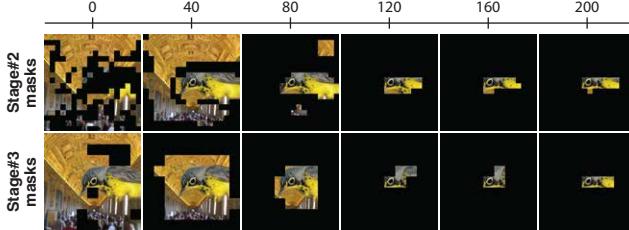


Figure 7. Visualization of the evolution of learned masks at two different stages (Stage #2 and Stage #3) of a ConvNeXt-tiny+AIM [2, 25%] model throughout the training epochs. As training progresses, the masks gradually become more sparse and accurately localized, highlighting the model’s improved ability to identify and focus on regions containing genuine features in a self-supervised manner.

Table 2. **Lower performance with Bottom-up Guiding Approach.** Result of the bottom-up ConvNeXt-t+AIM applying the masking mechanism in the second and third stages in the vanilla ConvNeXt-tiny, compared to the top-down Refocs+ConvNeXt-t model.

Model	CUB-200 (%)
bottom-up masking [1, 25%] ConvNeXt-t	72.79 ( $\pm 8.51$ )
ConvNeXt-t+AIM [1, 25%]	<b>88.82 (<math>\pm 0.213</math>)</b>
bottom-up masking [2, 25%] ConvNeXt-t	84.00 ( $\pm 1.38$ )
ConvNeXt-t+AIM [2, 25%]	<b>88.677 (<math>\pm 0.25</math>)</b>

a clear “what you see causes what you get” relationship between features and predictions.

## 6. Analysis and Ablation

The following further explores the effectiveness of AIM.

### 6.1. Top-down approach v/s Bottom-up approach

Inspired by [49], we initially tested a bottom-up masking approach, utilizing the same mask estimators described in Section 3 but without a top-down pathway. In this setup, each convolutional stage of the backbone model had two branches: the original convolutional path and a mask estimator. The latter predicted a binary mask, applied to the convolutional output to create spatially sparse feature maps that proceeded to the next stage. Unlike [49], we did not employ a skip-connection to convert sparse feature maps back into dense ones, aiming to preserve inherent explainability. However, this bottom-up guiding method performed poorly on the CUB-200 dataset, as shown in Table 2, where bracketed numbers indicate the used stage and annealing. Furthermore, the generated masks tended to remain fully active despite applying the mask active-area loss (see Appendix D.4 for further analysis), unlike the naturally focused masks produced by the top-down approach.

### 6.2. Does AIM have a center bias?

To investigate potential center bias [8] in our experiments, we tested models on images with birds positioned at the edges rather than center-frame. Table 3 shows

Table 3. **AIM does not exploit the center-bias** AIM manages to detect and focus on the bird achieving higher results compared to the vanilla ConvNeXt-tiny model

Model	CUB-200 (%)
ConvNeXt-tiny	76.98 ( $\pm 0.18$ )
ConvNeXt-t+AIM [1, 25%]	<b>79.33 (<math>\pm 0.45</math>)</b>

that while both vanilla ConvNeXt-tiny and ConvNeXt-tiny+AIM experienced performance decreases compared to center-cropped images, AIM still outperformed the baseline by approximately 2.5%. Furthermore, as depicted in Figure 8, AIM continued to generate masks focused on birds despite partial visibility, confirming our approach does not depend on center bias for its effectiveness.



Figure 8. **AIM models do not have a center-bias.** This illustration shows the merged masks generated by ConvNeXt-t+AIM (2, 25%) on two images from CUB-200.

## 7. Conclusion

In this work, we propose Amending Inherent Interpretability via Self-Supervised Masking (AIM), a simple yet effective method that encourages networks to focus on dependable rather than spurious features via a self-supervised feature-masking process. Evaluated using the Energy Pointing Game (EPG) score on out-of-distribution and fine-grained classification tasks, AIM improves localization on dependable features without sacrificing accuracy or requiring annotations beyond class labels. AIM produces inherently interpretable models by integrating sparse, top-down feature selection directly into the forward pass. It consistently improves both accuracy and localization across diverse datasets and architectures, with minimal overhead. These results highlight AIM’s potential as a lightweight and scalable approach to training models that are robust, generalizable, and aligned with meaningful visual cues.

**Future Work.** We plan to extend AIM to Vision Transformers [6] by either reshaping patch embeddings into spatial feature maps or leveraging the hierarchical structure of Swin-Transformers [23] for more seamless integration.

## 8. Acknowledgment.

Funded in part by the DFG (German Research Foundation, RTG 2853/1). S.A. and M.K. acknowledge support by DFG Research Unit 5336 Learning2Sense.

## References

- [1] Shashank Agnihotri, Shashank Priyadarshi, Hendrik Sommerhoff, Julia Grabinski, Andreas Kolb, and Margret Keuper. Roll the dice: Monte carlo downsampling as a low-cost adversarial defence, 2024. 15
- [2] Saeid Asgari, Aliasghar Khani, Fereshte Khani, Ali Gholami, Linh Tran, Ali Mahdavi Amiri, and Ghassan Hamarneh. Masktune: Mitigating spurious correlations by forcing to explore. *Advances in Neural Information Processing Systems*, 35:23284–23296, 2022. 2
- [3] Moritz Böhle, Mario Fritz, and Bernt Schiele. B-cos networks: Alignment is all we need for interpretability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10329–10338, 2022. 2
- [4] Aditya Chattopadhyay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pages 839–847. IEEE, 2018. 2
- [5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 5
- [6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. 8
- [7] Ryan Farrell. Cub-200-2011 segmentations, 2022. 14, 15
- [8] Mishal Fatima, Steffen Jung, and Margret Keuper. Corner cases: How size and position of objects challenge imagenet-trained models. In *Synthetic Data for Computer Vision Workshop@ CVPR 2025*, 2025. 8
- [9] Thomas Fel, Ivan F Rodriguez Rodriguez, Drew Linsley, and Thomas Serre. Harmonizing the object recognition strategies of deep neural networks with humans. *Advances in neural information processing systems*, 35:9432–9446, 2022. 2
- [10] Siddhartha Gairola, Moritz Böhle, Francesco Locatello, and Bernt Schiele. How to probe: Simple yet effective techniques for improving post-hoc explanations, 2025. 2
- [11] Yuyang Gao, Tong Steven Sun, Guangji Bai, Siyi Gu, Sung-soo Ray Hong, and Zhao Liang. Res: A robust framework for guiding visual explanation. In *proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 432–442, 2022. 2
- [12] Yuyang Gao, Tong Steven Sun, Liang Zhao, and Sung-soo Ray Hong. Aligning eyes between humans and deep neural network through interactive attention alignment. *Proceedings of the ACM on Human-Computer Interaction*, 6(CSCW2):1–28, 2022. 2
- [13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 5
- [14] Robin Hesse, Simone Schaub-Meyer, and Stefan Roth. Content-adaptive downsampling in convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4544–4553, 2023. 2, 3, 4
- [15] J Hoffmann, S Agnihotri, Tonmoy Saikia, and Thomas Brox. Towards improving robustness of compressed cnns. In *ICML Workshop on Uncertainty and Robustness in Deep Learning (UDL)*, 2021. 15
- [16] Jeremy Howard. Imagewoof: a subset of 10 classes from imagenet that aren’t so easy to classify, 2019. 15
- [17] Peng-Tao Jiang, Chang-Bin Zhang, Qibin Hou, Ming-Ming Cheng, and Yunchao Wei. Layercam: Exploring hierarchical class activation maps for localization. *IEEE Transactions on Image Processing*, 30:5875–5888, 2021. 2
- [18] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022. 1, 2
- [19] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pages 5338–5348. PMLR, 2020. 2, 5, 15, 31
- [20] Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Unmasking clever hans predictors and assessing what machines really learn. *Nature communications*, 10(1):1096, 2019. 1
- [21] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2017. 3, 4, 20
- [22] Drew Linsley, Dan Shiebler, Sven Eberhardt, and Thomas Serre. Learning what and where to attend. *arXiv preprint arXiv:1805.08819*, 2018. 2
- [23] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021. 8
- [24] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s, 2022. 3, 5
- [25] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. 13
- [26] Masahiro Mitsuhashi, Hiroshi Fukui, Yusuke Sakashita, Takanori Ogata, Tsubasa Hirakawa, Takayoshi Yamashita, and Hironobu Fujiyoshi. Embedding human knowledge into deep neural network via attention map. *arXiv preprint arXiv:1905.03540*, 2019. 2
- [27] Mazda Moayeri, Sahil Singla, and Soheil Feizi. Hard imagenet: Segmentations for objects with strong spurious cues, 2022. 5, 7, 15
- [28] Suzanne Petryk, Lisa Dunlap, Keyan Nasseri, Joseph Gonzalez, Trevor Darrell, and Anna Rohrbach. On guiding visual attention with language specification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18092–18102, 2022. 1, 2, 5, 14, 18, 32
- [29] Harish Guruprasad Ramaswamy et al. Ablation-cam: Visual explanations for deep convolutional network via gradient-free localization. In *proceedings of the IEEE/CVF winter*

- conference on applications of computer vision*, pages 983–991, 2020. 2
- [30] Sukrut Rao, Moritz Böhle, Amin Parchami-Araghi, and Bernt Schiele. Studying how to efficiently and effectively guide models with explanations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1922–1933, 2023. 2, 18
- [31] Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. Right for the right reasons: Training differentiable models by constraining their explanations. *arXiv preprint arXiv:1703.03717*, 2017. 2
- [32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 15
- [33] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019. 1, 5, 6, 14, 29, 30
- [34] Patrick Schramowski, Wolfgang Stammer, Stefano Teso, Anna Brugger, Franziska Herbert, Xiaoting Shao, Hans-Georg Luigs, Anne-Katrin Mahlein, and Kristian Kersting. Making deep neural networks right for the right scientific reasons by interacting with their explanations. *Nature Machine Intelligence*, 2(8):476–486, 2020. 2
- [35] Thomas A Sebeok and Robert Ed Rosenthal. The clever hans phenomenon: Communication with horses, whales, apes, and people. *Annals of the New York Academy of Sciences*, 1981. 1
- [36] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 2, 5, 15
- [37] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, 2019. 5, 15
- [38] Haifeng Shen, Kewen Liao, Zhibin Liao, Job Doornberg, Maoying Qiao, Anton Van Den Hengel, and Johan W Verjans. Human-ai interactive and continuous sensemaking: A case study of image classification using scribble attention maps. In *extended abstracts of the 2021 CHI conference on human factors in computing systems*, pages 1–8, 2021. 2
- [39] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMIR, 2017. 2
- [40] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2014. 15
- [41] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017. 2
- [42] Saeid Asgari Taghanaki, Aliasghar Khani, Fereshte Khani, Ali Gholami, Linh Tran, Ali Mahdavi-Amiri, and Ghassan Hamarneh. Masktune: Mitigating spurious correlations by forcing to explore, 2022. 2, 3
- [43] Damien Teney, Ehsan Abbasnedjad, and Anton van den Hengel. Learning what makes a difference from counterfactual examples and gradient supervision. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*, pages 580–599. Springer, 2020. 2
- [44] Stefano Teso. Toward faithful explanatory active learning with self-explainable neural nets. In *Proceedings of the Workshop on Interactive Adaptive Learning (IAL 2019)*, pages 4–16. CEUR Workshop Proceedings, 2019. 2
- [45] Stefano Teso and Kristian Kersting. Explanatory interactive machine learning. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 239–245, 2019. 2
- [46] Stefano Teso, Öznur Alkan, Wolfgang Stammer, and Elizabeth Daly. Leveraging explanations in interactive machine learning: An overview. *Frontiers in Artificial Intelligence*, 6:1066049, 2023. 2
- [47] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 5, 7, 15
- [48] Robert van der Klis, Stephan Alaniz, Massimiliano Mancini, Cassio F Dantas, Dino Ienco, Zeynep Akata, and Diego Marcos. Pdisconet: Semantically consistent part discovery for fine-grained recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1866–1876, 2023. 18
- [49] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2020. 2, 3, 8, 22, 34
- [50] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 2, 5, 14
- [51] Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-cam: Score-weighted visual explanations for convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 24–25, 2020. 2, 5, 15
- [52] Xinyue Xu, Yi Qin, Lu Mi, Hao Wang, and Xiaomeng Li. Energy-based concept bottleneck models: Unifying prediction, concept intervention, and probabilistic interpretations. In *The Twelfth International Conference on Learning Representations*, 2024. 18
- [53] Ziyan Yang, Kushal Kafle, Franck Dernoncourt, and Vicente Ordonez. Improving visual grounding by encouraging consistent gradient-based explanations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19165–19174, 2023. 2

- [54] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2018. [14](#), [15](#)

# **AIM: Amending Inherent Interpretability via Self-Supervised Masking**

## Paper #2932 Supplementary Material

### **Table of Contents**

The supplementary material covers the following information:

#### **A. Implementation Details**

- A.1 Hyperparameters
- A.2 Active-area Loss Threshold Annealing
- A.3 Datasets
  - A.3.1 CUB-200
  - A.3.2 WaterBirds
  - A.3.3 TraverlingBirds
  - A.3.4 ImageNet100
  - A.3.5 ImageWoof
  - A.3.6 ImageNetHard
- A.4 Testing the Center Bias of the Masks

#### **B. Quantitative Results**

- B.1 Center Bias
- B.2 Attribution-Agnostic Localization Performance
- B.3 Extended Results of ConvNeXt+AIM
- B.4 Comparison to Other Methods

#### **C. Qualitative Results**

- C.1 Additional Quantitative Results
- C.2 Center Bias
- C.3 Qualitative Comparison of Masks and Attribution Maps: AIM vs. Vanilla Backbone Models

#### **D. Ablation Results**

- D.1 Active-Area Loss Annealing
- D.2 Without the Auxiliary Losses
- D.3 Do We Need Multiple Mask Estimators at Each Level?
- D.4 The Bottom-Up Bottlenecking Approach
- D.5 Emphasizing peripheral regions in mask estimator initialization

### **A. Implementation Details**

In this section, we present a comprehensive overview of the implementation details of our proposed model to ensure reproducibility and facilitate future research. We begin by outlining the specific hyperparameters used during training, the datasets utilized in our experiments. We then detail the annealing procedures employed to adapt the threshold of the active-area loss used for the mask estimators in the AIM architecture. Additionally, we explain the shifted-center cropping technique implemented to assess the model’s susceptibility to center bias.

#### **A.1. Hyperparameters**

The primary hyperparameters used for our AIM models are the shown in Table 4:

Table 4. The Hyperparameters values used for training AIM models.

Hyperparameter	Value
Validation dataset ratio	20% of training dataset
Batch Size	512
top-down pathway learning rate	0.01
Weight Decay	0.001
RandAugment	(ops=3, magnitude=9)
Label Smoothing	0.05
Learning Rate Schedule	cosine
Optimizer	AdamW [25]
drop out rate	0.3
drop out path rate	0.3

We used different learning rates for each of the backbones, as listed in Table 5. Specifically, for the ConvNeXt-tiny+AIM model a much smaller learning rate, compared to those used for the other models, is needed for the training to converge.

Table 5. General training setting used for training AIM.

Model	backbone Learning rate
AIM+ConvNeXt	7e-6
ResNet50+AIM	0.001
ResNet101+AIM	0.001

## A.2. Computational Overhead

Tab. 6 quantifies the computational overhead of our proposed AIM module. The integration results in a marginal increase in both GFLOPs and model parameters, confirming the method’s efficiency.

Table 6. Comparison of GFLOPs and the Number of Parameters with Increment Over Baseline Models

Model Name	GFLOPs	Parameters (M)
ConvNeXt-tiny	4.5	28.0
ConvNeXt-tiny+AIM [1]	5.2 (+0.7)	30.7 (+2.7)
ConvNeXt-tiny+AIM [2]	4.6 (+0.1)	29.9 (+1.9)
ResNet50	4.1	23.9
ResNet50+AIM [2]	5.1 (+1.0)	27.6 (+3.7)
ResNet50+AIM [3]	4.4 (+0.3)	26.6 (+2.7)

## A.3. Active-area loss threshold annealing

We employ a masking annealing technique to ensure a seamless adaptation of the network to the masking process. The main idea of this technique is to increase the masking or decrease the number of active elements (elements with value 1 in the binary mask) as the training evolves (see Figure 9). This is done by controlling the active-area loss threshold throughout the training process, where we start with fully active masks (a threshold of 1.0), enabling the entire network to operate without constraints, and as training advances, we decrease that threshold with each epoch until reaching the final wanted value (for example 0.35 which means 35% of the mask is only active). The network then uses this final value for the rest of the training. This annealing technique aids the network in adjusting more effectively to the masking constraints, resulting in improved mask quality and a more stable learning process. The number of epochs or steps for which the annealing of the active-area threshold is carried out is treated as a hyperparameter.

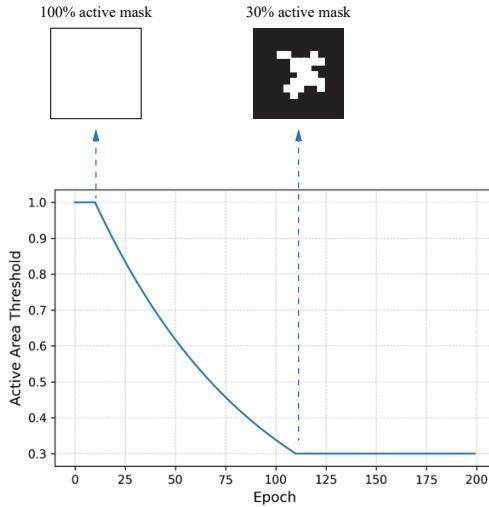


Figure 9. The annealing process of the active-area loss threshold begins either at the start of training or at a specific epoch (e.g., epoch 10 in the figure) and continues for a set number of epochs (e.g., 100 epochs, which is half of the total 200 training epochs). After this period, a final threshold of 0.3 is maintained for the remainder of the training.

## A.4. Datasets

In the following sections, we provide details on the datasets used in our experiments: CUB, Waterbirds-100%, Waterbirds-95%, and TravelingBird.

### A.4.1. CUB-200

The Caltech-UCSD Birds-200-2011 [50], known as CUB-200-2011, is one of the most well-known datasets in the fine-grained visual classification domain. The dataset consists of 11,788 images of 200 bird species, roughly divided in half for training and half for evaluation. There are an average of 30 images per class in the training dataset and 29 images per class in the test dataset. Throughout this work, we divided the primary training dataset into 80% and 20% training and validation datasets to search over hyperparameters. In our experiments, we utilized an input image size of  $(224 \times 224)$  pixels, and for computing the EPG scores, we relied on the binary segmentation [7] masks.

### A.4.2. WaterBirds

Waterbirds dataset is a synthetic dataset [33] designed to test classification models by introducing a controllable distribution shift, it is specifically engineered to assess how models respond to shifts in group distributions. The authors took the different bird species from the CUB-200-2011 dataset and simplified the task to be a two-class classification: Waterbirds and Landbirds. They manipulate the backgrounds using the binary segmentation masks [7], where they replaced the original background with either water or land scenes taken from the Places dataset [54]. This creates four different groups: Landbirds on land-background, Landbirds on water-background, Waterbirds on land-background, and Waterbirds on water-background.

The ability to control the construction of the dataset allows researchers to explore how models handle spurious correlations. In the training dataset, the majority of images fall into main groups - Landbirds on land and Waterbirds on water. However, in the validation and test datasets there is an equal distribution across the four groups, creating a deliberate distribution shift between training and test datasets.

The two mainly used versions of this dataset are, the Waterbirds100% version [28], which presents the most extreme challenge, with training dataset that have a perfect correlation between the type of the bird and the background-Water bird on Water background and Land bird on Land background. The other version is the Waterbirds95% [33] where 5% of the training images come from the out-of-distribution groups: Landbird on Water-background and Waterbird on land-background. In our experiments, we utilized an input image size of  $(224 \times 224)$  pixels. For computing the EPG scores, we also relied on the binary segmentation masks [7] of the CUB-200 dataset.

#### A.4.3. TravelingBirds

TravelingBirds dataset [19] is a variant of the CUB dataset and it is constructed in a similar way to that of Waterbirds dataset. While it preserves the original 200 classes of the CUB-200-2011 dataset, it changes the background of the birds to spuriously correlate the target label  $y$  and the image background within the training set only. The Authors used the binary segmentation masks to isolate the bird's pixels from its original background and put them onto a different background scene taken from the Places dataset [54]. Each bird species is laid out on a unique but randomly selected type of scene. During test time, the association between bird label and their background scene type is randomized, Completely disrupting the training set's correlation between background and class labels, resulting in a challenging adversarial setting. Following [19], we utilized an input image size of  $(299 \times 299)$  pixels, and for computing the EPG scores, we also relied on the binary segmentation masks [7] of the CUB-200 dataset.

#### A.4.4. ImageNet-100

ImageNet-100 [47] is a widely used [1, 15] subset of the full ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 dataset [32]. It is composed of 100 distinct object classes selected from the original 1000, containing approximately 128,000 training images and 5,000 validation images. The primary motivation for using this subset is to enable faster model training, hyperparameter tuning, and experimentation compared to the resource-intensive full dataset, while still offering a diverse and challenging multi-class classification benchmark. In our experiments, we follow standard practice and use an input image size of  $(224 \times 224)$  pixels.

#### A.4.5. ImageWoof

ImageWoof [16] is a challenging 10-class subset of ImageNet [32] designed for rapid yet difficult experimentation. The dataset is intentionally curated to be a hard, fine-grained classification problem, as it contains 10 breeds of dogs that are visually very similar. By focusing on these hard-to-distinguish classes, ImageWoof provides a computationally inexpensive benchmark that is more demanding than a random subset of ImageNet of a similar size. This makes it particularly useful for quickly iterating on and evaluating new model architectures and training methodologies. In our experiments, we utilize an input image size of  $(224 \times 224)$  pixels.

#### A.4.6. Hard-ImageNet

ImageNet-Hard [27] is a benchmark designed to evaluate if models classify images "for the right reasons" rather than relying on spurious correlations. It consists of 19k training images and 750 quintuply-validated test images across 15 classes: Dog sled, Howler Monkey, Seat Belt, Ski, Sunglasses, Swimming Cap, Balance Beam, Horizontal bar, Patio, Hockey Puck, Miniskirt, Space Bar, Volleyball, Baseball Player, and Snorkel.

The benchmark challenges models with images where the object of interest is captured under suboptimal conditions (e.g., not large or centered). Leveraging segmentation masks helps diagnose whether a model's prediction is based on the object itself or on misleading background cues, thus providing a deeper understanding of model behavior beyond simple accuracy metrics. For evaluation, we use an input size of  $(224 \times 224)$  pixels.

### A.5. Energy Pointing Game (EPG) Score

The Energy Pointing Game (EPG) score [51] is a metric designed to quantify the spatial localization capabilities of a model. Specifically, it evaluates the extent to which the model's attribution aligns with the ground-truth object regions.

The computation of the EPG score requires the following components:

- **Attribution Map:** An importance map generated by an attribution method such as Saliency Maps [40], GradCAM [36], Guided GradCAM [37], or any comparable technique.
- **Ground-Truth Binary Mask:** A binary segmentation mask that delineates the region corresponding to the object of interest.

The EPG score is calculated as the ratio between the total attribution energy within the active (foreground) region of the binary mask and the total attribution energy across the entire attribution map. A higher EPG score indicates that the model concentrates its decision-making evidence within the relevant object regions, thereby demonstrating better spatial grounding of its predictions.

### A.6. Testing the center bias of the masks

One of the concerns we had after seeing the masks that our method generated on the CUB-200 dataset was the susceptibility of our models' masks to center bias. Thus, rather than center-crop the images, we designed a cropping mechanism that, given

a specific deviation from the center, randomly chose one new center for cropping in a way that guarantees to violate the center bias of the object in the newly cropped image. We denote this modified dataset as the shifted-center CUB-200. Figure 10 illustrates the shifted cropping technique used to assess the center bias of the generated masks. Each colored dot represents the center of its corresponding square crop, which is outlined in the same color. For each image, one of the four centers is randomly selected, and the image is cropped accordingly.

It is important to note that this cropping technique might also crop important parts from the object of interest, making the classification much harder due to the lack of task-related features. For example, the body might be cropped out while only the legs of the bird are kept.



Figure 10. The image illustrates the shifted cropping technique used to assess the center bias of the generated masks. Each colored dot represents the center of its corresponding square crop, which is outlined in the same color. For each image, one of the four centers is randomly selected, and the image is cropped accordingly.

## B. Quantitative results

This section provides a comprehensive overview of the performance metrics obtained from our trained models under different configurations.

### B.1. Center bias

Table 7 presents the performance metrics for the baseline vanilla ConvNeXt-tiny model as well as the ConvNeXt-tiny+AIM models on the Shifted-center CUB-200 dataset. Notably, both the vanilla ConvNeXt-tiny and all variants of the AIM architecture experienced a performance drop, with the ConvNeXt-tiny showing a decrease of approximately 10% and the AIM variants showing a decline of around 9% compared to the center-cropped experiments (see Figure 1). Despite this reduction in accuracy, our proposed model variants still outperform the baseline ConvNeXt-tiny model by nearly 2%. For qualitative comparison of the GradCAM and the generated masks see Figure 12

Table 7. presents the performance metric scores for the baseline vanilla ConvNeXt-tiny model as well as the ConvNeXt-tiny+AIM models on the Shifted-center CUB-200 dataset. Notably, both the vanilla ConvNeXt-tiny and all variants of the AIM architecture experienced a performance drop, with the ConvNeXt-tiny showing a decrease of approximately 10% and the AIM variants showing a decline of around 9% compared to the center-cropped experiments (see Table 9). Despite this reduction in accuracy, our proposed model variants still outperform the baseline ConvNeXt-tiny model by nearly 2%. For qualitative comparison of the GradCAM and the generated masks see Figure 12.

Model	Test Accuracy ( $\pm$ Std)
ConvNeXt-tiny	76.98 ( $\pm$ 0.18)
AIM+ConvNeXt (1, 25%)	79.08 ( $\pm$ 0.44)
AIM+ConvNeXt (1, 35%)	79.33 ( $\pm$ 0.45)
AIM+ConvNeXt (1, No Annealing)	79.13 ( $\pm$ 0.35)
AIM+ConvNeXt (2, 25%)	79.11 ( $\pm$ 0.12)
AIM+ConvNeXt (2, 35%)	79.11 ( $\pm$ 0.12)
AIM+ConvNeXt (2, No Annealing)	79.12 ( $\pm$ 0.13)

Upon analyzing the generated masks on the cropped test images (see Figures 12), we can see that large bird regions are missing, which can attribute the performance reduction to the loss of essential parts during the cropping process. The masks clearly focus on the task-related regions or what's left of them after cropping. This is particularly evident in the masks generated by stage#2 in the second example, where the model focuses on the bird's legs and belly.

## B.2. Attribution-Agnostic Localization Performance:

In addition to GradCAM, we evaluate other attribution methods to assess the generality of our localization improvements. Table 8 reports EPG scores on the Waterbirds-95% dataset using Guided GradCAM and Guided Backprop, both of which show consistent gains with AIM.

Table 8. Attribution-agnostic EPG improvements on Waterbirds-95%. AIM consistently boosts localization scores regardless of the attribution method used.

Model	EPG score $\uparrow$	
	Guided GradCAM	Guided Backpropagation
Vanilla ConvNext-tiny	46.17( $\pm$ 2.5)	31.26 ( $\pm$ 5.4)
ConvNext+AIM (2, 25%)	<b>76.77 (<math>\pm</math> 2.7)</b>	<b>51.89 (<math>\pm</math> 0.3)</b>

## B.3. Extended Results of ConvNeXt+AIM:

In this section, we provide extended results for our AIM-enhanced ConvNeXt model, as detailed in Table 9. We supplement our findings on the Waterbirds dataset by including the overall accuracy metric, which further highlights the benefits of our approach. For instance, the best-performing AIM model achieves an overall accuracy of  $89.1\% \pm 0.8\%$  on the Waterbirds (100%) benchmark, a significant increase from the baseline's  $71.2\% \pm 2.2\%$ . Furthermore, we present results on the CUB-200 dataset, where AIM-based models also outperform the baseline model.

Table 9. Average Test Accuracies for different configurations of ConvNeXt+AIM. Comparison of the ConvNeXt-tiny baseline against our AIM-enhanced models on three benchmarks. Our method achieves substantial gains, most notably improving the **worst-group accuracy** on the Waterbirds dataset, which highlights its effectiveness in mitigating spurious correlations. All values are mean accuracy (%)  $\pm$  standard deviation.

Model	CUB			Waterbirds				Travelingbirds		
	Acc	EPG	WG-Acc	100% Overall Acc	EPG	WG-Acc	95% Overall Acc	EPG	Acc	EPG
ConvNeXt-t	87.9 ( $\pm$ 0.02)	68.3 ( $\pm$ 0.1)	39.6 ( $\pm$ 5.4)	71.2 ( $\pm$ 2.2)	57.19 ( $\pm$ 6)	81.6 ( $\pm$ 3.2)	94.8 ( $\pm$ 0.3)	68.3 ( $\pm$ 3.2)	59.5 ( $\pm$ 0.8)	74.4 ( $\pm$ 0.6)
ConvNeXt-t+AIM[1, 25%]	<b>88.6 (<math>\pm</math>0.1)</b>	<b>76.804 (<math>\pm</math>3.3)</b>	73.6 ( $\pm$ 4.5)	86.2 ( $\pm$ 2.6)	60.11 ( $\pm$ 1.3)	91.2 ( $\pm$ 0.8)	95.7 ( $\pm$ 0.7)	<b>77.1 (<math>\pm</math>5.15)</b>	77.1 ( $\pm$ 0.3)	79 ( $\pm$ 0.7)
ConvNeXt-t+AIM[1, 35%]	88.18 ( $\pm$ 0.1)	55 ( $\pm$ 3.8)	77.1 ( $\pm$ 4.4)	88.4 ( $\pm$ 1.8)	57.2 ( $\pm$ 1.3)	90.7 ( $\pm$ 0.7)	96 ( $\pm$ 0.3)	63 ( $\pm$ 1.2)	71.5 ( $\pm$ 1.3)	72.6 ( $\pm$ 1.5)
ConvNeXt-t+AIM[2, 25%]	87.78 ( $\pm$ 0.2)	75.17( $\pm$ 1.2)	74 ( $\pm$ 5)	84( $\pm$ 6)	58 ( $\pm$ 1.3)	<b>92.7 (<math>\pm</math>1.2)</b>	<b>96.6 (<math>\pm</math>0.2)</b>	75 ( $\pm$ 6)	<b>77.4 (<math>\pm</math>0.2)</b>	<b>85 (<math>\pm</math>2)</b>
ConvNeXt-t+AIM[2, 35%]	88.5 ( $\pm$ 0.2)	62.5 ( $\pm$ 0.3)	<b>78.1 (<math>\pm</math>2.3)</b>	<b>89.1 (<math>\pm</math>0.8)</b>	<b>68.5 (<math>\pm</math>3.6)</b>	92.3 ( $\pm$ 0.6)	96.31 ( $\pm$ 0.1)	71.7 ( $\pm$ 6.4)	71 ( $\pm$ 0.4)	77.7 ( $\pm$ 0.4)

## B.4. Comparison To Other Methods:

This section compares AIM to other methods. Unlike our approach, which does not use any form of guidance, other methods often rely on guidance mechanisms to direct the trained model. Examples of such guidance include using binary masks [30] or leveraging the output of a CLIP-based model controlled by a text prompt GALS [28] or using specific loss to account for imbalanced data [48]. Our proposed AIM solely relies on image labels used by traditional image classification methods.

**Table 10. Average Test Accuracies for Various Models with and without AIM Modification.** This table presents the average test accuracies (with standard deviations) for different models on the CUB-200, Waterbirds, and Travelingbirds datasets. The models include ConvNeXt-tiny, ResNet50, and ResNet101 with and without the AIM modification. The AIM-enhanced models are denoted as  $\{backbone\}+AIM$  (stage index, mask active-area loss), such as ConvNeXt-tiny+AIM(1, 25%). The results are organized by dataset and further divided into categories: overall accuracy and worst-group accuracy for both 100% and 95% subsets where applicable. Improvements or declines in performance due to the AIM modification are highlighted in green and red, respectively. This comprehensive comparison provides insights into the effectiveness of the AIM approach in enhancing model performance across different datasets and scenarios.

Model	CUB-200 (%)	Waterbirds (%)				Travelingbirds (%)	
		100%		95%			
		Worst-Group	Overall	Worst-Group	Overall		
ConvNeXt-tiny	87.9 ( $\pm 0.02$ )	39.6 ( $\pm 5.4$ )	71.2 ( $\pm 2.2$ )	81.6 ( $\pm 3.2$ )	94.8 ( $\pm 0.3$ )	59.5 ( $\pm 0.8$ )	
<b>ConvNeXt-t+AIM[1, 25%]</b>	<b>88.6 (<math>\pm 0.1</math>)</b>	73.6 ( $\pm 4.5$ )	86.2 ( $\pm 2.6$ )	91.2 ( $\pm 0.8$ )	95.7 ( $\pm 0.7$ )	77.1 ( $\pm 0.3$ )	
<b>ConvNeXt-t+AIM[1, 35%]</b>	88.18 ( $\pm 0.1$ )	77.1 ( $\pm 4.4$ )	88.4 ( $\pm 1.8$ )	90.7 ( $\pm 0.7$ )	96 ( $\pm 0.3$ )	71.5 ( $\pm 1.3$ )	
<b>ConvNeXt-t+AIM[2, 25%]</b>	87.78 ( $\pm 0.2$ )	74 ( $\pm 5$ )	84 ( $\pm 6$ )	<b>92.7 (<math>\pm 1.2</math>)</b>	<b>96.6 (<math>\pm 0.2</math>)</b>	<b>77.4 (<math>\pm 0.2</math>)</b>	
<b>ConvNeXt-t+AIM[2, 35%]</b>	88.5 ( $\pm 0.2$ )	<b>78.1 (<math>\pm 2.3</math>)</b>	<b>89.1 (<math>\pm 0.8</math>)</b>	92.3 ( $\pm 0.6$ )	96.31 ( $\pm 0.1$ )	71 ( $\pm 0.4$ )	
DRO [48] + ResNet50	-	-	-	91.4 ( $\pm 1.1$ )	-	-	
GALS [28] + ResNet50	-	<b>56.71</b> (-)	-	76.54 (-)	-	-	
BCos-ResNet50 [30]	-	56.1 ( $\pm 4$ )	-	-	-	-	
ResNet50	81.32	41.29 ( $\pm 1.99$ )	69.48 ( $\pm 2$ )	71.09 ( $\pm 5.98$ )	91.21 ( $\pm 0.27$ )	50.21 ( $\pm 0.5$ )	
<b>ResNet50+AIM (2, 25%)</b>	83.90	52.19 ( $\pm 9.11$ )	73.84 ( $\pm 1.14$ )	75.97 ( $\pm 2.344$ )	92.1 ( $\pm 0.31$ )	65.2 ( $\pm 0.24$ )	
<b>ResNet50+AIM (3, 25%)</b>	83.16	40.12 ( $\pm 2.1$ )	71.36 ( $\pm 1.7$ )	77.2875 ( $\pm 0.404$ )	92.38 ( $\pm 0.2$ )	64 ( $\pm 0.51$ )	
ECBM [52] + ResNet101	-	-	-	-	-	<b>58.4</b> (-)	
ResNet101	76.41 ( $\pm 0.619$ )	36.48 ( $\pm 4.22$ )	71.69 ( $\pm 2.2$ )	77.565 ( $\pm 3.2$ )	92.82 ( $\pm 0.31$ )	19.54 ( $\pm 0.34$ )	
<b>ResNet101+AIM (2, 25%)</b>	82.55 ( $\pm 0.22$ )	38.11 ( $\pm 1.4$ )	71.46 ( $\pm 0.53$ )	82.39 ( $\pm 0.88$ )	93.12 ( $\pm 0.74$ )	44.03 ( $\pm 0.96$ )	
<b>ResNet101+AIM (3, 25%)</b>	82.635 ( $\pm 0.28$ )	38.81 ( $\pm 1.4$ )	71.8 ( $\pm 0.69$ )	82.39 ( $\pm 1.328$ )	93.34 ( $\pm 0.7$ )	45.13 ( $\pm 0.7$ )	

## C. Qualitative results

In this section, we explore the qualitative aspects of our study by showcasing masks and Grad-CAM attributions across different settings. These visualizations provide insights into the models’ decision-making processes, highlighting areas of focus and variation in response to different configurations.

### C.1. Additional qualitative results:

In the figure 11, we illustrate additional examples from CUB, waterbirds-100%, Travelingbirds and Hard-ImageNet datasets, where it is clear from the GradCAM attributes that AIM-based models focus on the object and ignore the spurious features.

### C.2. Center bias

Figure 12 presents a qualitative comparison of different architectural configurations of AIM on the Shifted-center CUB-200 setting. The figure illustrates the masks generated at each stage for two primary architectural variants: ConvNeXt-tiny+AIM (1), shown in the top group of images, and ConvNeXt-tiny+AIM (2), shown in the bottom group. Each variant employs different mask active-area thresholds, with each row representing a distinct threshold setting. The masks produced by AIM (columns denoted stage#1 masks, stage#2 masks, stage#3 masks) explicitly delineate the regions utilized by the model at each stage, thereby demonstrating that AIM effectively mitigates susceptibility to center bias. The “Merged Masks” column demonstrates where the final feature maps will be zero, highlighting the discarded regions.

### C.3. Qualitative Comparison of Masks and Attribution Maps: AIM vs. Vanilla Backbone Models

In this section, we present qualitative results to illustrate the effectiveness of our proposed approach. We selected five different input images from each of the following datasets: CUB-200, Waterbirds-95%, Waterbirds-100%, and Traveling Birds. For these images, we compare the GradCAM attribution maps generated by the AIM+[backbone] models with different mask

active-area thresholds to those produced by the vanilla [backbone] models. Additionally, we display the generated merged masks from the AIM+[backbone] models.

- For the **CUB-200** dataset we show the ConvNeXt-tiny+AIM ( $T=1$ ) and ( $T=2$ ) outputs in Figure 13 and Figure 14 respectively, while we illustrate the output of ResNet50+AIM ( $T=2$ ) and ( $T=3$ ) Figure 15 and Figure 16 respectively.
- For the **Waterbirds-95%** dataset, we present the ConvNeXt-tiny+AIM outputs with mask active-area thresholds ( $T=1$ ) and ( $T=2$ ) in Figure 17 and Figure 18, respectively. Additionally, we illustrate the outputs of ResNet50+AIM with ( $T=2$ ) and ( $T=3$ ) in Figure 19 and Figure 20, respectively.
- For the **Waterbirds-100%** dataset, we show the outputs of ConvNeXt-tiny+AIM and ResNet50+AIM in Figure 23 and Figure 24, respectively.
- For the **TravelingBirds** dataset, we present the outputs of ResNet50+AIM and ResNet101+AIM in Figure 22 and Figure 21, respectively.

## D. Ablation results

In this appendix section, we conduct an ablation study to explore the roles of individual components and parameters within our proposed method. By systematically adjusting or removing certain elements, we aim to gain a better understanding of their contributions to the model’s performance.

### D.1. Active-Area Loss Annealing:

This section analyzes the performance variations in ConvNeXt-tiny+AIM models, focusing on two architectural variants: ConvNeXt-tiny+AIM ( $T=1$ ) and ConvNeXt-tiny+AIM ( $T=2$ ). The analysis examines changes in accuracy and energy pointing game (EPG) scores relative to the vanilla ConvNeXt-tiny model when the mask active-area threshold  $\tau$  is adjusted on the Waterbrids-95% dataset.

The results in Table 11 reveal that reducing the active-area threshold leads to improvements in both the worst group and overall accuracies; this can also be seen in Figure 25. This finding suggests that by constraining the spatial areas upon which the models depend, they are compelled to identify and concentrate on regions pertinent to the task. This focus is evidenced by the higher EPG scores achieved, indicating effective task-related region identification.

Table 11. This table presents the performance changes, in terms of accuracy and energy pointing game (EPG) scores on the Waterbirds-95% dataset, for ConvNeXt-tiny+AIM models with two architectural variants: ConvNeXt-tiny+AIM ( $T=1$ ) and ConvNeXt-tiny+AIM ( $T=2$ ). These are compared against the vanilla ConvNeXt-tiny model when altering the mask active-area threshold  $\tau$ . The results indicate that a smaller active-area threshold leads to increases in both the worst group and overall accuracies. This suggests that limiting the spatial areas the models rely on encourages them to identify and focus on task-related regions, as reflected by high EPG scores. Another notable trend is that the ConvNeXt-tiny+AIM ( $T=1$ ) variants achieve better EPG scores, implying that the masks in this variant, applied across all three stages of the top-down pathway, effectively concentrate on the regions of interest (ROI). This is also seen more clearly in Figure 25.

Model	Annealing final threshold $\tau$	Waterbirds-95%		
		Worst-group	Overall	EPG
Vanilla ConvNeXt-tiny	–	79.63 ( $\pm$ )	93.755 ( $\pm 1.340$ )	63.52
AIM+ConvNeXt (1)	10%	92.16 ( $\pm$ )	95.62 ( $\pm$ )	84.3
AIM+ConvNeXt (2)		94 ( $\pm$ )	96.63 ( $\pm$ )	81
AIM+ConvNeXt (1)	15%	92.24 ( $\pm$ )	95.93 ( $\pm$ )	81.7
AIM+ConvNeXt (2)		93.33 ( $\pm$ )	96.53 ( $\pm$ )	62.3
AIM+ConvNeXt (1)	20%	91.46 ( $\pm$ )	95.69 ( $\pm$ )	73
AIM+ConvNeXt (2)		93.53 ( $\pm$ )	96.51 ( $\pm$ )	57.4
AIM+ConvNeXt (1)	25%	91.74 ( $\pm$ )	96.31 ( $\pm$ )	74.22
AIM+ConvNeXt (2)		93.24 ( $\pm$ )	95.62 ( $\pm$ )	72.1
AIM+ConvNeXt (1)	30%	91.35 ( $\pm$ )	96.36 ( $\pm$ )	59
AIM+ConvNeXt (2)		92.87 ( $\pm$ )	96.65 ( $\pm$ )	49
AIM+ConvNeXt (1)	35%	91.11 ( $\pm$ )	96.38 ( $\pm$ )	56.13
AIM+ConvNeXt (2)		92.59 ( $\pm$ )	96.39 ( $\pm$ )	45.83
AIM+ConvNeXt (1)	40%	90.68 ( $\pm$ )	96.24 ( $\pm$ )	41.7
AIM+ConvNeXt (2)		91.89 ( $\pm$ )	96.29 ( $\pm$ )	53.8

## D.2. Without The Auxiliary Losses:

As discussed in Section 3, the AIM architecture incorporates a classification loss as an auxiliary loss at each stage in the top-down pathway, following the approach from [21]. These losses help align the learned features at each stage with the final task. However, this raises an important question: How would the AIM network perform if these auxiliary losses were removed? What impact would this have on the network’s overall performance?

Table 12 examines the impact of removing auxiliary losses on the performance of ConvNeXt-tiny+AIM and ResNet50+AIM models across the CUB-200 and Waterbirds-95% datasets. The results, measured in terms of accuracy and energy pointing game (EPG) scores, reveal notable performance fluctuations when auxiliary losses are omitted. On the Waterbirds-95% dataset, some settings show performance gains, while others experience declines. Conversely, on the CUB-200 dataset, performance consistently deteriorates without the auxiliary losses. These observations highlight the critical role of auxiliary losses in maintaining stable and reliable performance across different models and datasets.

Table 12. **AIM networks performance worsens without auxiliary losses.** This table presents the performance variations of ConvNeXt-tiny+AIM and ResNet50+AIM on the CUB-200 and Waterbirds-95% datasets, measured in terms of accuracy (averaged over 4 different runs) and energy pointing game (EPG) scores. Red arrows indicate a decrease in the score compared to the corresponding architecture with the auxiliary losses, while green arrows represent an increase in scores. The results indicate that removing the auxiliary losses leads to fluctuations in performance across the datasets and models' variations. Specifically, on the Waterbirds-95% dataset, performance increases for some settings while decreasing for others. On the CUB-200 dataset, performance consistently worsens compared to when auxiliary losses are used. These findings underscore the importance of auxiliary losses in achieving consistent performance across models and datasets.

Model	$\tau$	Auxiliary losses	CUB-200			Waterbirds-95%	
			Acc	EPG	Worst-group Acc	Overall	EPG
ConvNeXt-tiny+AIM (1, $\tau$ )	25%	no	88.29 $\downarrow$ ( $\pm 0.16$ )	59.4 $\uparrow$	92.24 $\uparrow$ ( $\pm 0.73$ )	96.84 $\uparrow$ ( $\pm 0.086$ )	73.82 $\downarrow$
	25%	yes	88.63 ( $\pm 0.13$ )	58.54	91.31 ( $\pm 1.1$ )	96.77 ( $\pm 0.1$ )	74.22
	35%	no	88.51 $\downarrow$ ( $\pm 0.075$ )	50.12 $\downarrow$	91.7 $\uparrow$ ( $\pm 0.76$ )	96.70 $\uparrow$ ( $\pm 0.35$ )	69.57 $\uparrow$
	35%	yes	88.82 ( $\pm 0.21$ )	57.49	90.1 ( $\pm 1.44$ )	96.63 $\pm 0.13$ )	56.13
	no annealing	no	88.6 $\downarrow$ ( $\pm 0.15$ )	44.69 $\downarrow$	92.23 $\uparrow$ ( $\pm 2.88$ )	94.89 $\downarrow$ ( $\pm 2.13$ )	48.17 $\uparrow$
	no annealing	yes	88.77 ( $\pm 0.1$ )	53.56	89.3 ( $\pm 2.48$ )	96.07 ( $\pm 0.16$ )	42.57
	25%	no	88.31 $\downarrow$ ( $\pm 0.19$ )	59.2 $\downarrow$	93.18 $\uparrow$ ( $\pm 1.1$ )	97.27 $\uparrow$ ( $\pm 0.32$ )	78.4 $\uparrow$
	25%	yes	88.55 ( $\pm 0.3$ )	60.27	90.1 ( $\pm 1.46$ )	96.43 ( $\pm 0.2$ )	72.12
ConvNeXt-tiny+AIM (2, $\tau$ )	35%	no	88.47 $\downarrow$ ( $\pm 0.17$ )	54.19 $\downarrow$	92.01 $\uparrow$ ( $\pm 0.51$ )	96.85 $\uparrow$ ( $\pm 0.1$ )	63.98 $\uparrow$
	35%	yes	88.67 ( $\pm 0.25$ )	56.82	91.41 ( $\pm 0.22$ )	96.40 ( $\pm 0.23$ )	45.83
	no annealing	no	88.68 $\uparrow$ ( $\pm 0.15$ )	41.49 $\downarrow$	91.78 $\uparrow$ ( $\pm 0.74$ )	95.94 $\downarrow$ ( $\pm 0.26$ )	58.48 $\downarrow$
	no annealing	yes	88.62 ( $\pm 0.21$ )	55.54	88.7 ( $\pm 1.4$ )	96.39 ( $\pm 0.104$ )	65.84
	25%	no	79.74 $\downarrow$ ( $\pm 0.14$ )	55.86 $\downarrow$	77.68 $\downarrow$ ( $\pm 0.673$ )	92.41 $\downarrow$ ( $\pm 0.184$ )	61.89 $\downarrow$
ResNet50+AIM (2, $\tau$ )	25%	yes	80.9 ( $\pm 0.345$ )	57.56	91.31 ( $\pm 1.1$ )	96.77 ( $\pm 0.1$ )	74.22
	35%	no	79.85 $\downarrow$ ( $\pm 0.07$ )	54.54 $\downarrow$	77.41 $\downarrow$ (0)	92.35 $\uparrow$ ( $\pm 0.1$ )	51.66 $\downarrow$
	35%	yes	80.9 ( $\pm 0.12$ )	57	90.1 ( $\pm 1.44$ )	92.34 ( $\pm 0.25$ )	56.13
	no annealing	no	79.27 $\downarrow$ ( $\pm 0.23$ )	47.49 $\downarrow$	75.50 $\downarrow$ ( $\pm 0.545$ )	91.7 $\downarrow$ ( $\pm 0.26$ )	43.04 $\uparrow$
	no annealing	yes	80.84 ( $\pm 0.28$ )	53.81	89.3 ( $\pm 2.48$ )	91.9 ( $\pm 0.11$ )	42.57
ResNet50+AIM (3, $\tau$ )	25%	no	79.69 $\downarrow$ ( $\pm 0.35$ )	57.66 $\uparrow$	77.83 $\downarrow$ ( $\pm 0.61$ )	92.64 $\downarrow$ ( $\pm 0.38$ )	71.87 $\uparrow$
	25%	yes	80.9 ( $\pm 0.345$ )	56.76	91.31 ( $\pm 1.1$ )	96.77 ( $\pm 0.1$ )	61.89
	35%	no	79.77 $\downarrow$ ( $\pm 0.41$ )	55.04 $\uparrow$	78.38 $\downarrow$ ( $\pm 1.48$ )	91.92 $\downarrow$ ( $\pm 0.35$ )	46.35 $\downarrow$
	35%	yes	80.9 ( $\pm 0.12$ )	54.80	90.1 ( $\pm 1.44$ )	92.42 ( $\pm 0.46$ )	52.14
	no annealing	no	79.55 $\downarrow$ ( $\pm 0.21$ )	48.73 $\downarrow$	74.87 $\downarrow$ ( $\pm 2.0$ )	92.32 $\downarrow$ ( $\pm 0.32$ )	36.79 $\downarrow$
	no annealing	yes	80.84 ( $\pm 0.28$ )	52.71	89.3 ( $\pm 2.48$ )	87.39 ( $\pm 8.56$ )	41.78

### D.3. Do we need multiple mask estimators, one at each level?

One of the building blocks of our proposed method is the use of mask estimators in the top-down pathway, where we employ a mask estimator module at each stage, which we denote here as the Full AIM model. Given that in convolutional neural networks the feature maps from the last level determine the final semantic features used in the classification layer, a question arises: *can we rely solely on the mask generated at the first stage of the top-down pathway (this stage corresponds to the final convolutional layer in the backbone network used)?*

This question is motivated by the semantic richness of the feature maps at this level; thus, selecting areas by masking in these feature maps is supposed to truly represent the task-related semantics.

To investigate this question, we adjusted our network to use only the masks generated at the first stage of the top-down pathway. This is done by passing the first stage's masks to the subsequent stage after up-scaling them by a factor of 2, as the subsequent stage has higher spatial resolutions.

Table 13. Test accuracies of different AIM models on the CUB-200 dataset with masks only adapted from the first stage of the top-down pathway (highlighted with ✓ mark) versus the standard AIM network, which employs a mask estimator at each stage of the top-down pathway.

Model	One Mask Estimator	CUB-200 (%)
AIM+ConvNeXt (1, 25%)	✗	88.635 ( $\pm 0.13$ )
AIM+ConvNeXt (1, 35%)	✗	<b>88.82 (<math>\pm 0.213</math>)</b>
AIM+ConvNeXt (1, No Annealing)	✗	<u>88.775 (<math>\pm 0.044</math>)</u>
AIM+ConvNeXt (2, 25%)	✗	88.557 ( $\pm 0.296$ )
AIM+ConvNeXt (2, 35%)	✗	88.677 ( $\pm 0.25$ )
AIM+ConvNeXt (2, No Annealing)	✗	88.622 ( $\pm 0.211$ )
AIM+ConvNeXt (1, 25%)	✓	88.14 ( $\pm 0.44$ )
AIM+ConvNeXt (1, 35%)	✓	88.22 ( $\pm 0.26$ )
AIM+ConvNeXt (1, No Annealing)	✓	88.25 ( $\pm 0.33$ )
AIM+ConvNeXt (2, 25%)	✓	88.52 ( $\pm 0.22$ )
AIM+ConvNeXt (2, 35%)	✓	88.41 ( $\pm 0.20$ )
AIM+ConvNeXt (2, No Annealing)	✓	88.26 ( $\pm 0.28$ )

Table 13 presents the test accuracies of different configurations of our AIM models on the CUB-200 dataset. In these experiments, we compare the standard AIM network, which employs a mask estimator at each stage of the top-down pathway, with a modified version that uses masks only from the first stage of the top-down pathway. The modified approach passes the first-stage masks to subsequent stages to be applied on the corresponding feature maps. These models are indicated with a ✓ in the ‘One Mask Estimator’ column in the Table 13.

Analyzing the results, we observe that the models’ results are very close to each other, with models using masks solely from the first stage generally achieving slightly lower test accuracies than those employing mask estimators at each stage.

In convolutional neural networks, as input data moves through the network layers, the feature maps become richer in semantic information but lose spatial localization. We hypothesize that applying masks generated at the first stage of the top-down pathway, which is coarser and less localized (See Figure 26) compared to those from later stages, to subsequent lower stages could inadvertently include non-task-related regions, potentially diminishing performance, especially on out-of-distribution datasets.

#### D.4. The bottom-up bottlenecking approach:

One of the most relevant works to ours is [49], which uses a bottom-up masking approach to focus on essential regions in the generated feature maps. This approach relies on employing mask units between the main blocks of a convolutional network to produce binary masks highlighting task-relevant regions for the network. These mask units utilize the Gumbel-softmax trick to generate binary masks, which are then used to spatially bottleneck feature maps produced after each main block.

However, as discussed in the related work Section 2, this bottom-up approach needs skip-connections to work (See Figure 27) as the network needs more information to be flown from the shallow layer to the deepest layers to form an understanding of the scene, this compromised the inherent interpretability that comes naturally with the sparse feature maps, which with the skip-connections fall back to dense features maps.

We closely follow the method described in [49], but with one key difference: we utilized *fully sparse feature maps without skip connections*. This decision aligns with our objective of creating an interpretable model by focusing on generating fully sparse feature maps, which makes the decision-making process transparent (See Figure 28). For this purpose, we adhered to the same training setup we used for our approach, 200 epochs with active mask annealing applied for half the training period. We denote this architecture as *bottom-up* AIM Network.

We tested three architectural variants that differed in the number of masked convolutional blocks and experimented with two annealing threshold settings.

Table 14. This table presents the performance of Bi-gatedNet with ConvNeXt-tiny on the CUB-200 dataset when different blocks are masked and various mask active-area thresholds ( $\tau$ ) are applied. Masking all three blocks resulted in the lowest accuracy with high variability, indicating unstable training. Masking only the last two blocks improved accuracy but yielded poorly localized masks. Masking only the last block achieved the highest accuracy, surpassing the unmasked ConvNeXt-tiny model, and produced highly focused and localized masks.

Model Name	Bottom-Up Blocks Masked	CUB-200 (%)
ConvNext-tiny	-	86.827 ( $\pm 0.761$ )
standard ConvNeXt-tiny+AIM ( $T=1, \tau=25\%$ )	-	<b>88.82 (<math>\pm 0.213</math>)</b>
standard ConvNeXt-tiny+AIM ( $T=2, \tau=25\%$ )	-	<u>88.677 (<math>\pm 0.25</math>)</u>
<i>Bottom-up</i> ConvNeXt-tiny+AIM ( $\tau=25\%$ )	1, 2, 3	72.79 ( $\pm 8.51$ )
<i>Bottom-up</i> ConvNeXt-tiny+AIM ( $\tau=25\%$ )	2, 3	82.53 ( $\pm 1.39$ )
<i>Bottom-up</i> ConvNeXt-tiny+AIM ( $\tau=25\%$ )	3	86.202 ( $\pm 0.44$ )
<i>Bottom-up</i> ConvNeXt-tiny+AIM ( $\tau=35\%$ )	1, 2, 3	67.45 ( $\pm 8.52$ )
<i>Bottom-up</i> ConvNeXt-tiny+AIM ( $\tau=35\%$ )	2, 3	84.00 ( $\pm 1.38$ )
<i>Bottom-up</i> ConvNeXt-tiny+AIM ( $\tau=35\%$ )	3	86.975 ( $\pm 0.34$ )

Our main observation was that applying masks to all three blocks resulted in the lowest accuracy with high variability across trials (see Table 14), indicating unstable training and the model’s inability to learn effective representations. Additionally, the masks generated in this configuration (see Figure 29) showed almost no masking despite the application of the mask active-area loss with annealing.

When we adjusted the architecture to mask only the last two blocks, the accuracy improved sharply, by 10% for the annealing threshold of 0.25 and 13% for 0.35, reaching above 80% accuracy (see Table 14). However, the quality of the masks remained poor: the masks at the lower levels were still almost fully active, and only the mask from the last block focused on the bird.

This is evident in Figure 29, which shows that even when we limit the active area region to 25% and 35% using mask active-area loss annealing, the masks remain fully activated except for the one from the last block.

Furthermore, table 15 compares between our top-down and bottom-up approach in terms of sparsity and localization (EPG); the top-down method outperforms the bottom-up baseline on both metrics even without any sparsity loss (No annealing) on Waterbirds-95% dataset.

Table 15. Comparison of our top-down approach (ConvNext+AIM) against a bottom-up baseline on the Waterbirds-95% dataset. Our method demonstrates superior performance across all metrics, highlighting its ability to improve localization (EPG) and sparsity even without an explicit sparsity objective (No annealing).

Model	Worst group ACC ( $\pm$ std)	$\uparrow$ EPG ( $\pm$ std)	$\uparrow$ Sparsity score ( $\pm$ std)
Bottom-up ([1, 2, 3], 25%)	85.63 ( $\pm 4.2$ )	29.28 ( $\pm 21.4$ )	100 ( $\pm 0$ )
ConvNext+AIM (1, 25%)	<b>92.21 (<math>\pm 1.2</math>)</b>	<b>77.1 (<math>\pm 0.06</math>)</b>	17.7 ( $\pm 0.6$ )
Bottom-up ([1, 2, 3], No annealing)	87.41 ( $\pm 4.2$ )	24.66 ( $\pm 9.4$ )	100 ( $\pm 0$ )
ConvNext+AIM (1, No annealing)	<b>89.5 (<math>\pm 4.13</math>)</b>	<b>43.43 (<math>\pm 5.13</math>)</b>	71.48 ( $\pm 0$ )

In summary, training with the bottom-up approach indicates that masking fewer blocks leads to better performance and more robust masks.

## D.5. Emphasizing peripheral regions in mask estimator initialization

During training, the layers of the mask estimators are typically initialized randomly. To investigate the effect of the initialization scheme on the performance of the mask estimators, we propose an alternative initialization method that emphasizes the peripheral regions over the central regions. This is achieved by weighting the convolutional filter weights according to their spatial distance from the center of the filter kernel. Specifically, we assign larger initial values to the weights corresponding to the edges of the filters than to those closer to the center, effectively biasing the filters to be more responsive to features in the outer areas of the input (see Figure 30 for an illustration).

However, our experiments on the CUB-200 dataset reveal that this specialized initialization does not have a significant impact on performance. As shown in Table 16, the model’s accuracy remains essentially unchanged compared to when using standard random initialization. This suggests that the mask estimators are robust to the initial weight distribution and that the network is capable of learning effective representations regardless of this specific initialization strategy.

**Table 16. Emphasizing Peripheral Regions in Mask Initialization Does Not Impact AIM Network Performance.** This figure compares the AIM network’s performance on the CUB-200 dataset using two different initialization strategies for the mask estimators: standard random initialization and an initialization that emphasizes peripheral regions by assigning larger weights to filter edges. The results indicate that the model’s accuracy remains essentially unchanged between the two approaches. This suggests that the mask estimators are robust to the initial weight distribution, and the network can effectively learn meaningful representations regardless of this specific initialization strategy.

Model	CUB-200 (%)
<i>Edge-emphasized initialization</i> ConvNeXt-tiny+AIM (2, 25%)	<b>88.76 (<math>\pm 0.171</math>)</b>
<i>Standard random initialization</i> ConvNeXt-tiny+AIM (2, 25%)	88.678 ( $\pm 0.251$ )

## D.6. User Perception Study Details

To quantitatively assess if our model’s improved focus is perceptually meaningful, we conducted an online user study. Participants accessing the study were first directed to a welcome page that outlined the research goals, the task procedure, and our data privacy policy (Fig. 31). The introduction defined attribution maps as heatmaps used to visualize an AI’s focus and assured participants that all responses would be anonymous and confidential.

The study was designed to be completed in approximately 10 minutes. To achieve this while still covering a broad set of 100 images from the Waterbirds-100% dataset, the images were randomly partitioned into four disjoint subsets of 25. Each of the 107 participants was randomly assigned to evaluate exactly one of these four subsets. Following this assignment, participants proceeded to the main evaluation task, which consisted of 25 forced-choice comparison trials.

In each trial, participants were shown an image from their assigned subset, displayed alongside two GradCAM attribution maps: one from the baseline vanilla ConvNeXt and one from our AIM-equipped model. For each pair, they were asked to select the map that, in their opinion, “*more accurately and clearly focuses on the main object in the image, and less on irrelevant parts*”. An example of this trial layout is shown in Fig. 32. To mitigate positional bias, the on-screen placement of the two maps was randomized for every trial.

A total of 107 participants completed the study, yielding  $107 \times 25 = 2675$  individual evaluations. The results show a strong and significant user preference for our model. AIM’s attribution maps were chosen in **70.7%** of the evaluations. A two-sided binomial test confirms that this outcome is highly significant ( $p < 0.00001$ ), providing strong evidence that the explanations generated by AIM are more aligned with human intuition regarding object-centric focus compared to the baseline.

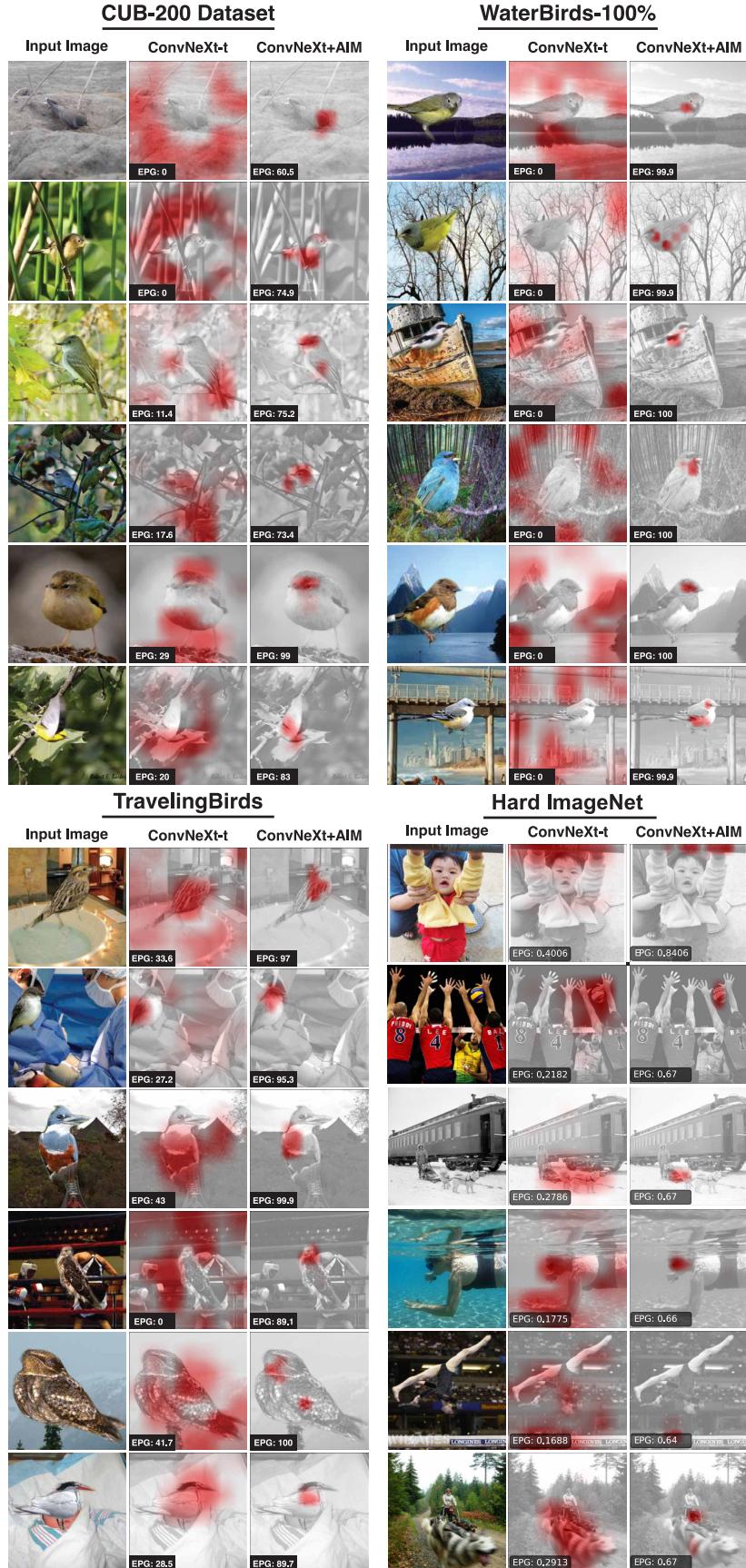
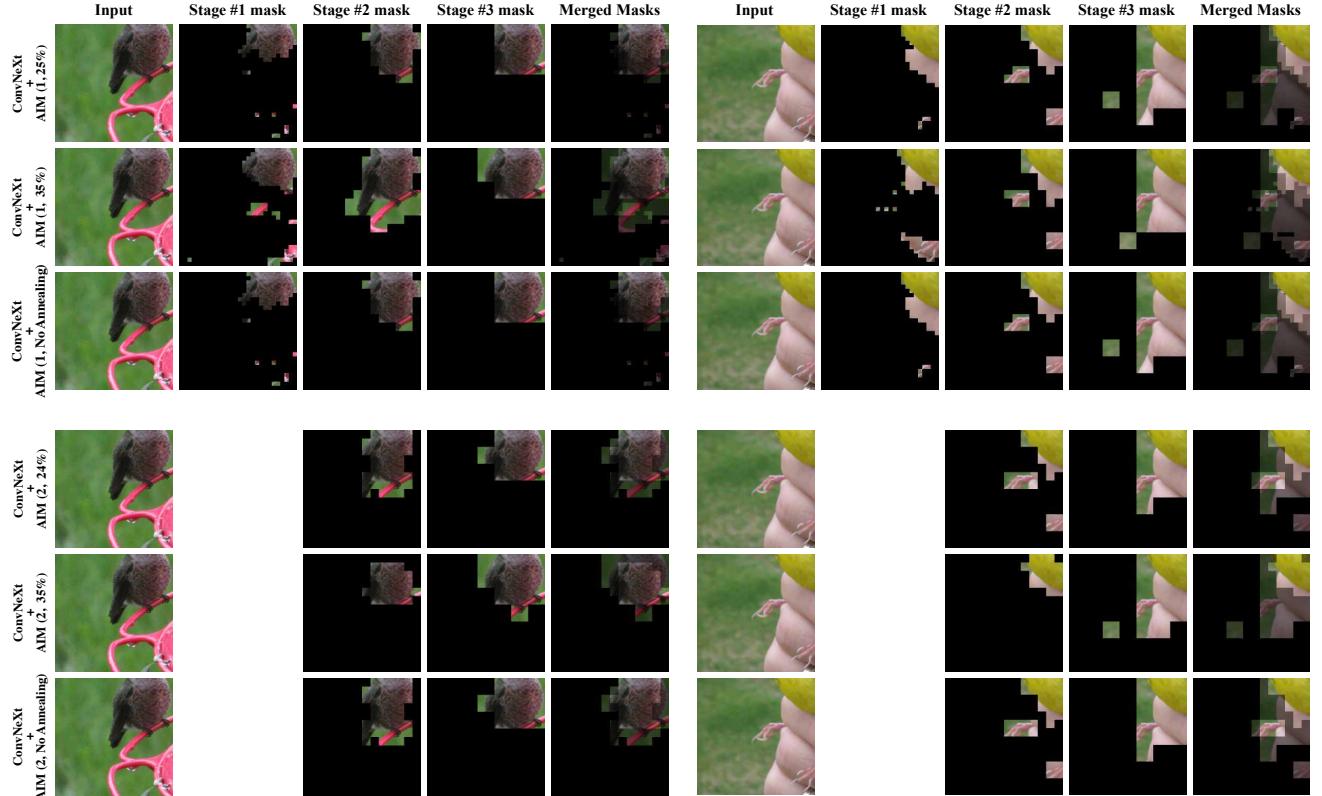


Figure 11. Models amended with AIM consistently exhibit enhanced localization of genuine features, effectively suppressing spurious cues in both in-domain and out-of-domain scenarios. A qualitative visualization of Gr25CAM heatmaps comparing baseline ConvNeXt-tiny models and ConvNeXt-tiny+AIM models across CUB-200, TravelingBirds, WaterBirds-100%, and Hard-ImageNet. The EPG scores are indicated on each heatmap.



**Figure 12. Qualitative Comparison between the different architectural configurations of AIM on the Shifted-center CUB-200 setting.** This figure illustrates the masks generated at each stage for two primary architectural variants: ConvNeXt+AIM ( $T=1$ ) (The group of images at the top) and ConvNeXt+AIM ( $T=2$ ) (the group of images at the bottom), each utilizing different mask active-area thresholds (each row represent different active-area loss setting). The masks produced by AIM (columns denoted stage#1 masks, stage#2 masks, stage#3 masks) explicitly delineate the regions utilized by the model at each stage, thereby demonstrating that AIM effectively mitigates susceptibility to center bias. The "Merged Masks" column demonstrates where the final feature maps will be zero, highlighting the discarded regions.

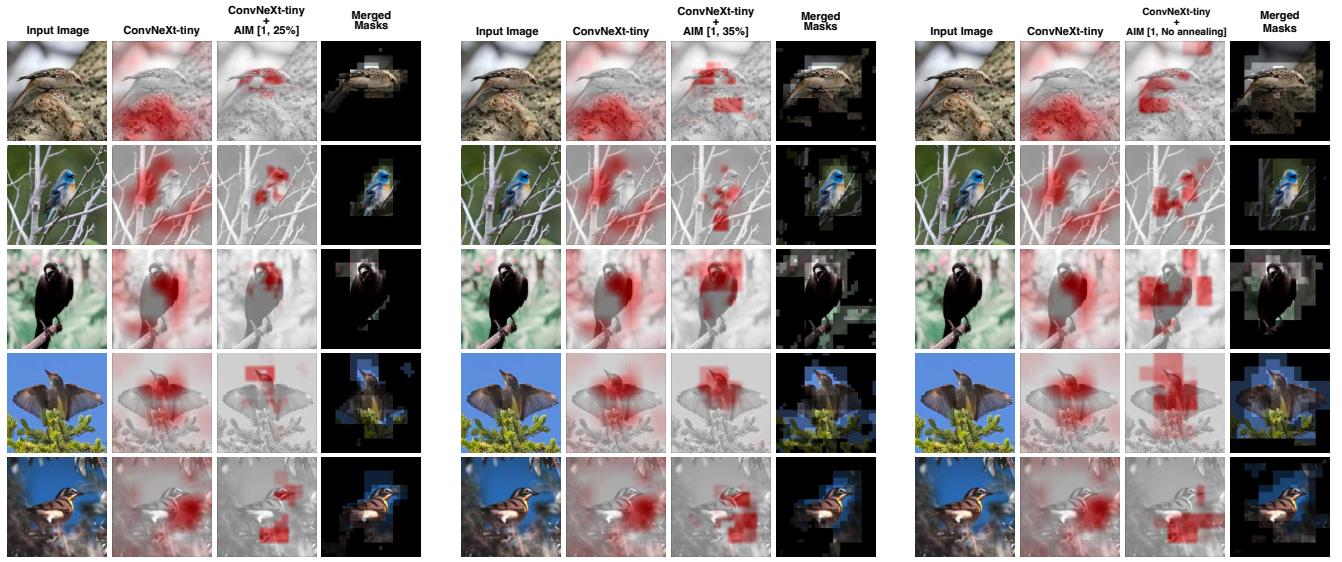


Figure 13. **Qualitative results on the CUB-200 dataset.** Comparison of GradCAM attribution maps between ConvNeXt-tiny+AIM models with different mask active-area thresholds, along with the generated merged masks.

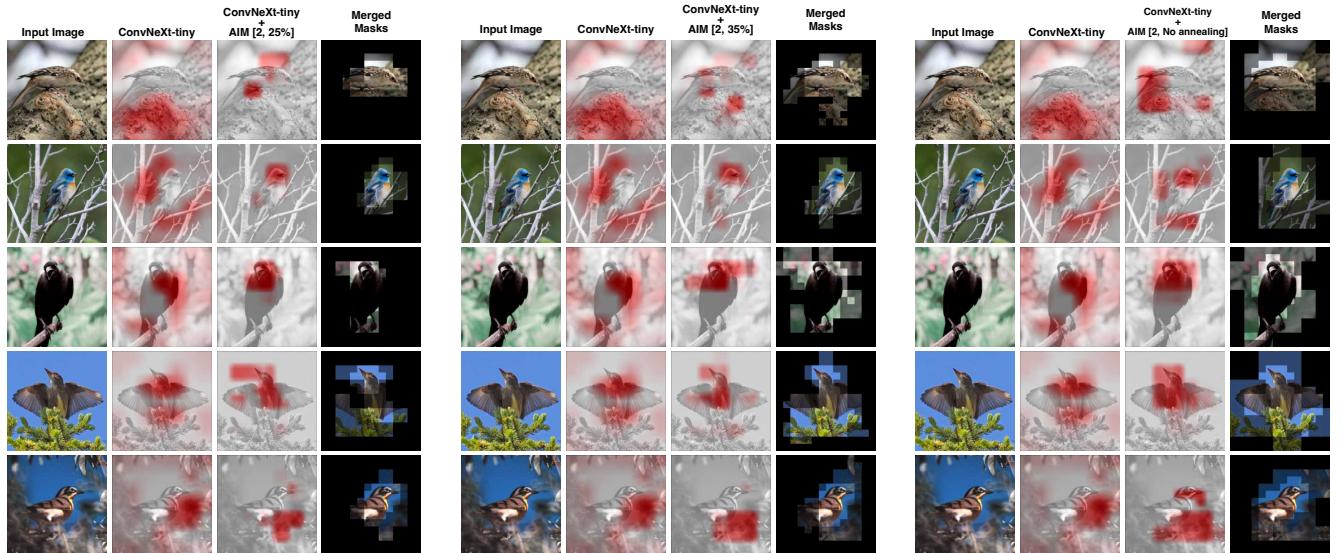
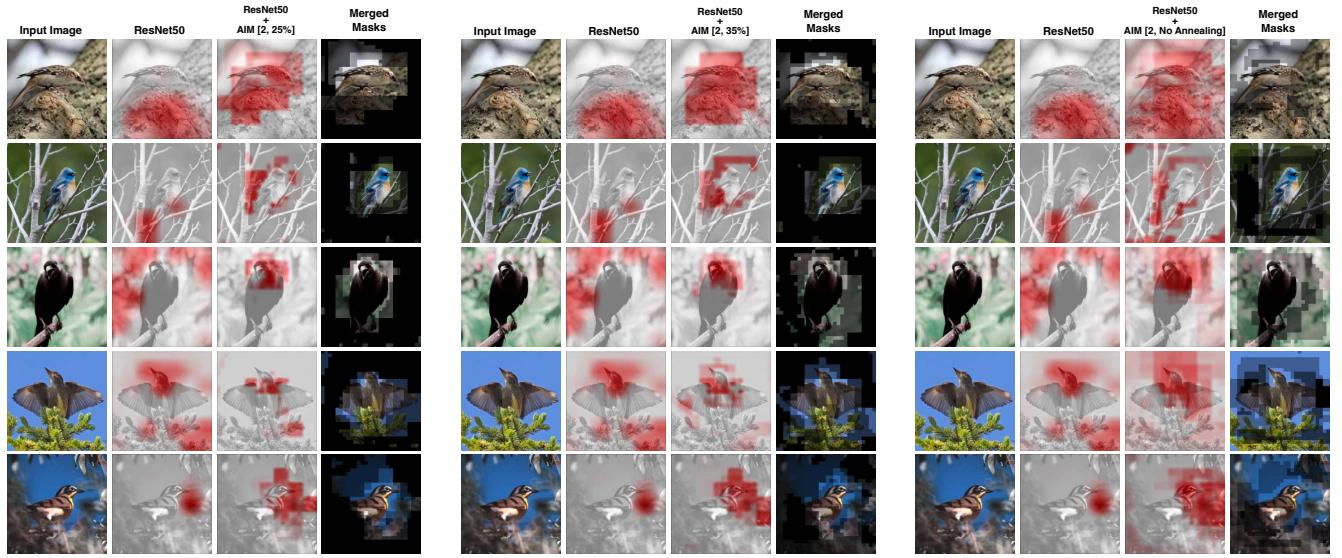
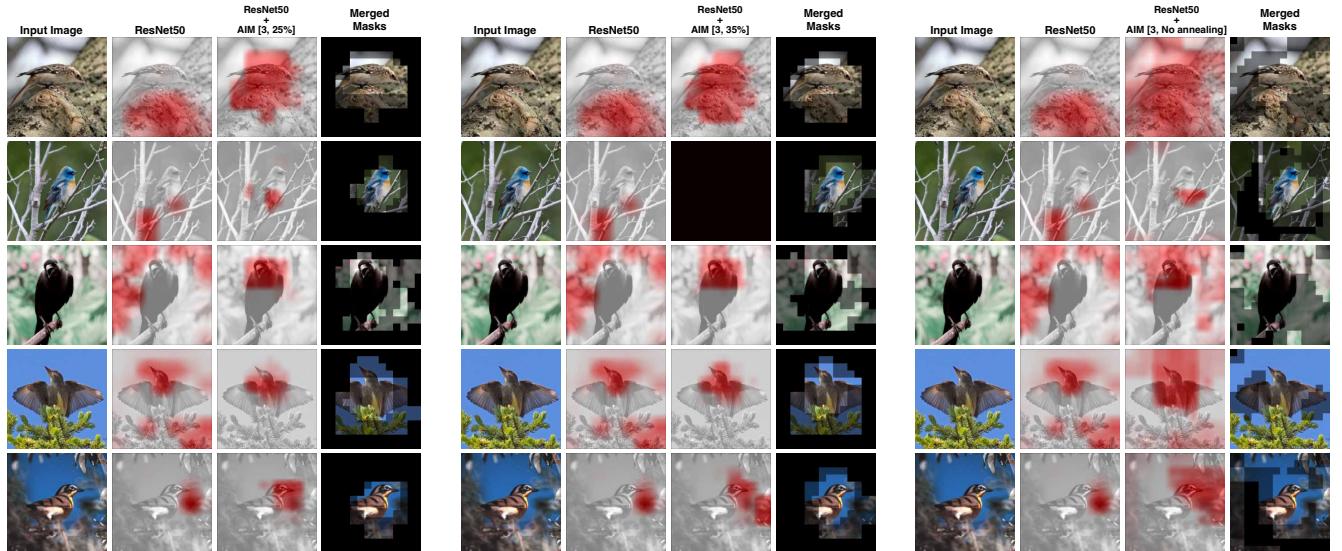


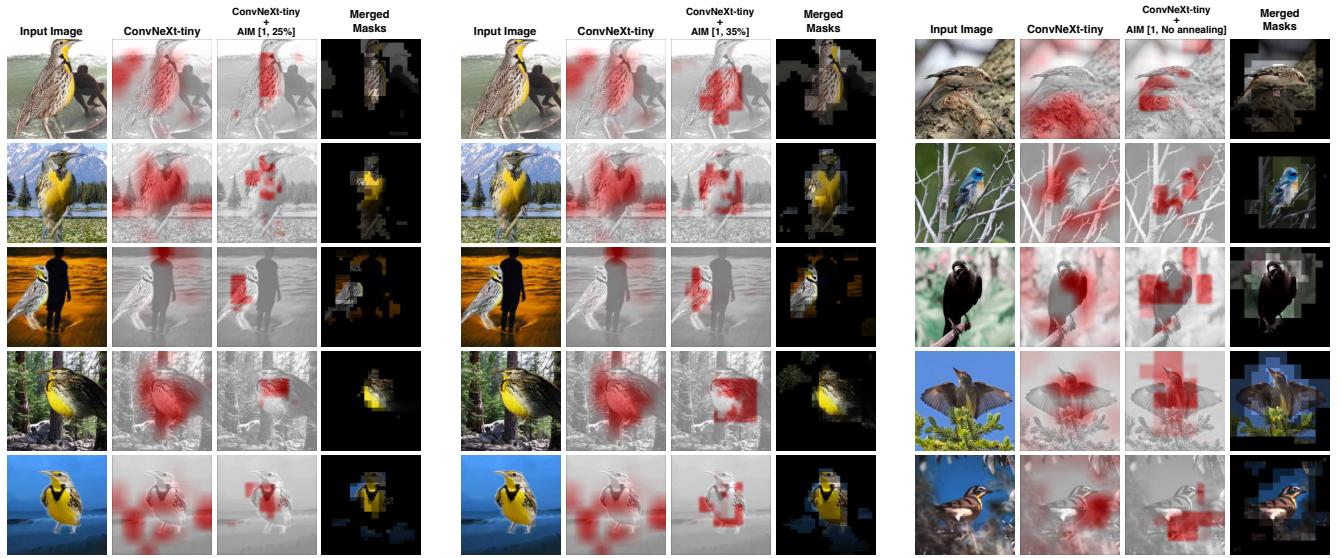
Figure 14. **Qualitative results on the CUB-200 dataset.** Comparison of GradCAM attribution maps between ConvNeXt-tiny+AIM models with different mask active-area thresholds, along with the generated merged masks.



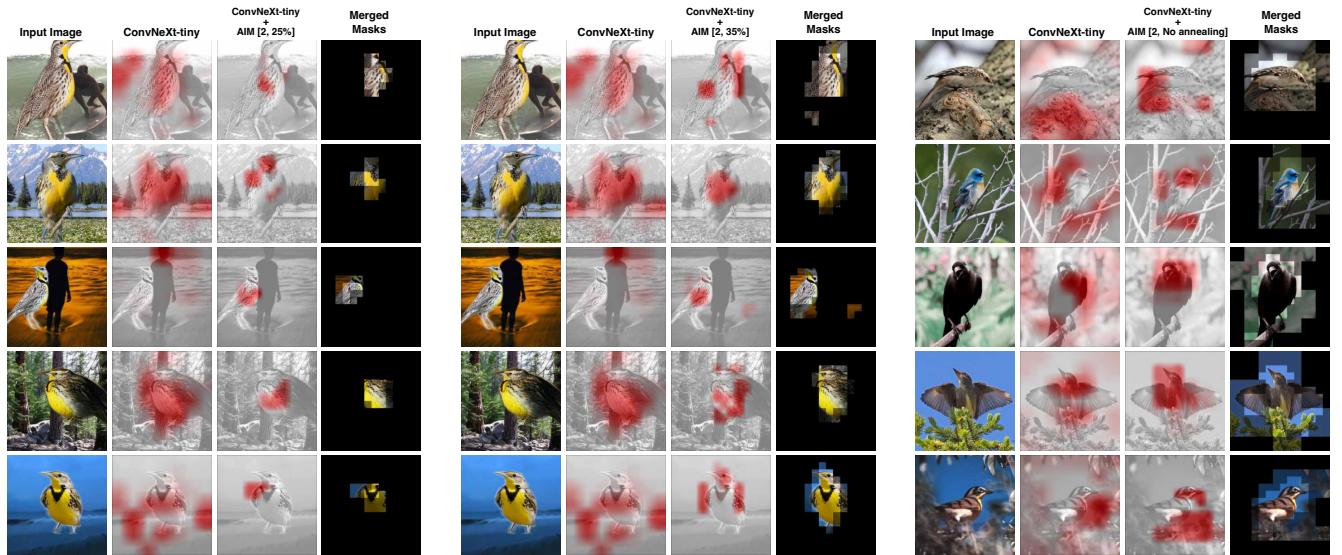
**Figure 15. Qualitative results on the CUB-200 dataset.** Comparison of GradCAM attribution maps between ResNet50+AIM models with different mask active-area thresholds, along with the generated merged masks.



**Figure 16. Qualitative results on the CUB-200 dataset.** Comparison of GradCAM attribution maps between ResNet50+AIM models with different mask active-area thresholds, along with the generated merged masks.



**Figure 17. Qualitative results on the Waterbirds-95% dataset [33].** Comparison of GradCAM attribution maps between ConvNeXt-tiny+AIM models with different mask active-area thresholds, along with the generated merged masks.



**Figure 18. Qualitative results on the Waterbirds-95% dataset [33].** Comparison of GradCAM attribution maps between ConvNeXt-tiny+AIM models with different mask active-area thresholds, along with the generated merged masks.

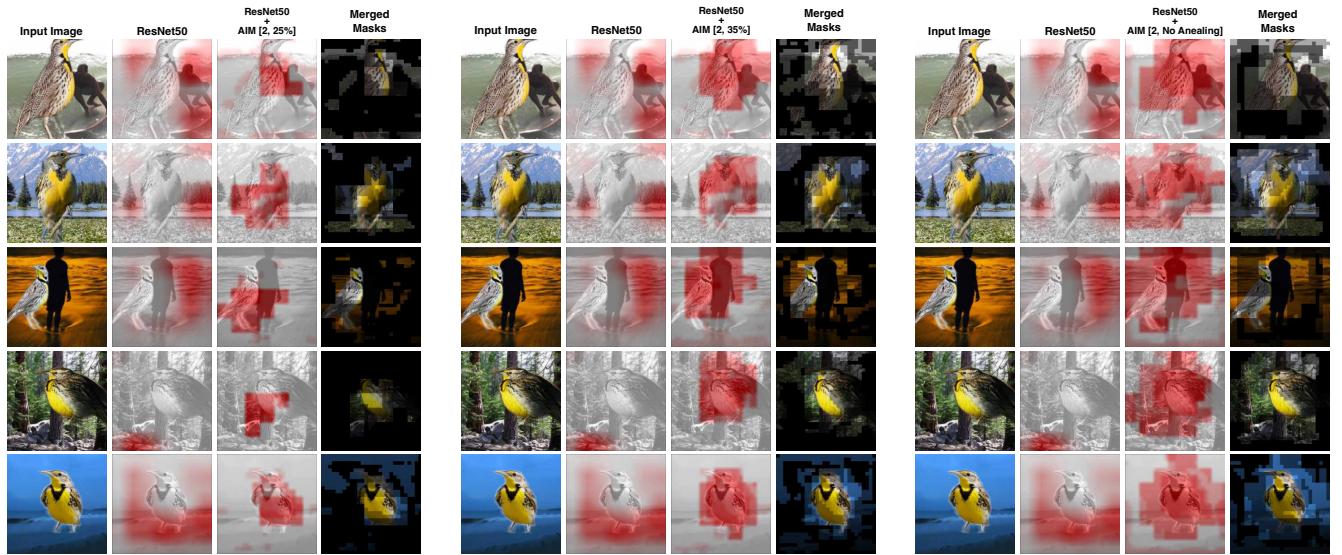


Figure 19. **Qualitative results on the Waterbirds-95% dataset [33].** Comparison of GradCAM attribution maps between ResNet50+AIM models with different mask active-area thresholds, along with the generated merged masks.

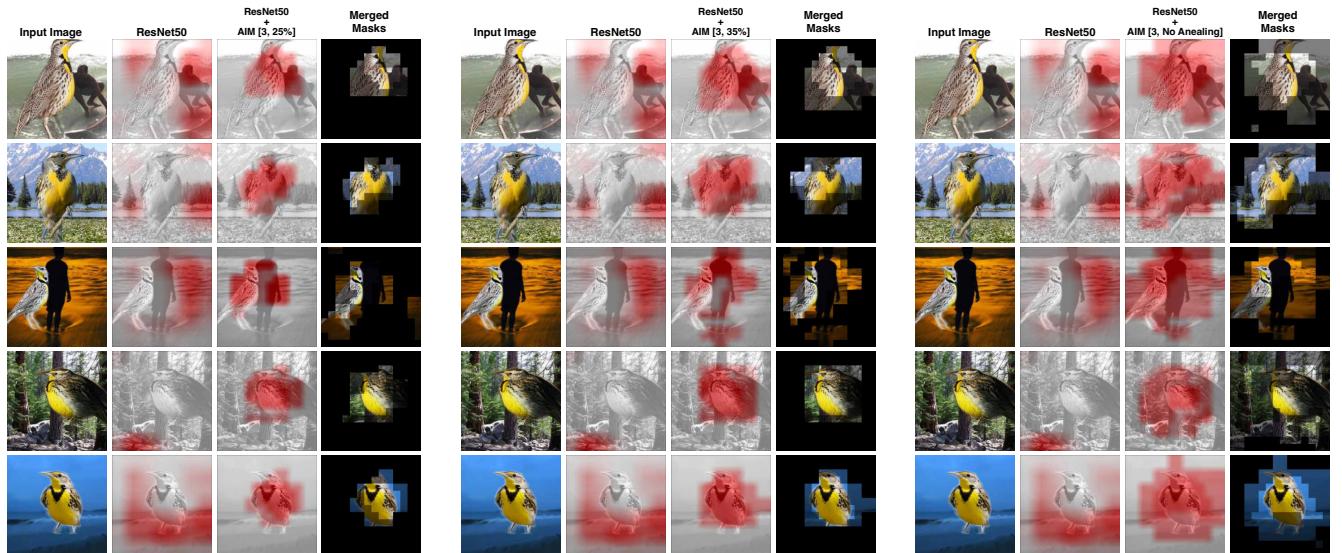


Figure 20. **Qualitative results on the Waterbirds-95% dataset [33].** Comparison of GradCAM attribution maps between ResNet50+AIM models with different mask active-area thresholds, along with the generated merged masks.



Figure 21. **Qualitative results on the Travelingbirds dataset [19].** Comparison of GradCAM attribution maps between ConvNeXt-tiny+AIM models with different mask active-area thresholds, along with the generated merged masks.

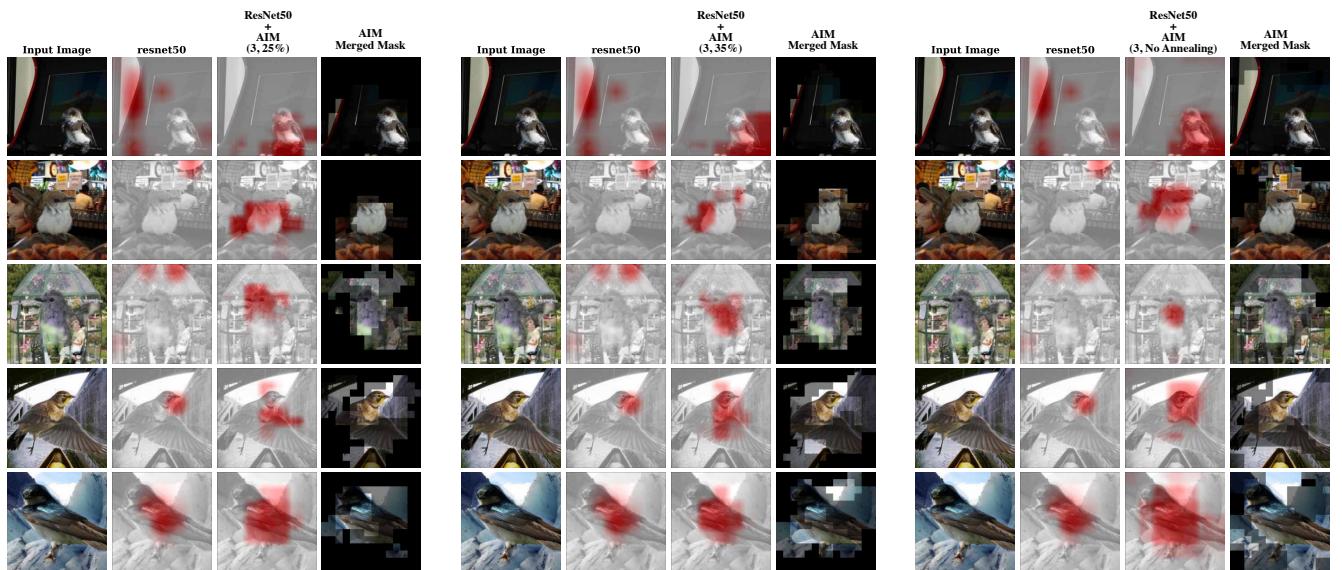


Figure 22. **Qualitative results on the Travelingbirds dataset [19].** Comparison of GradCAM attribution maps between ResNet50+AIM models with different mask active-area thresholds, along with the generated merged masks.

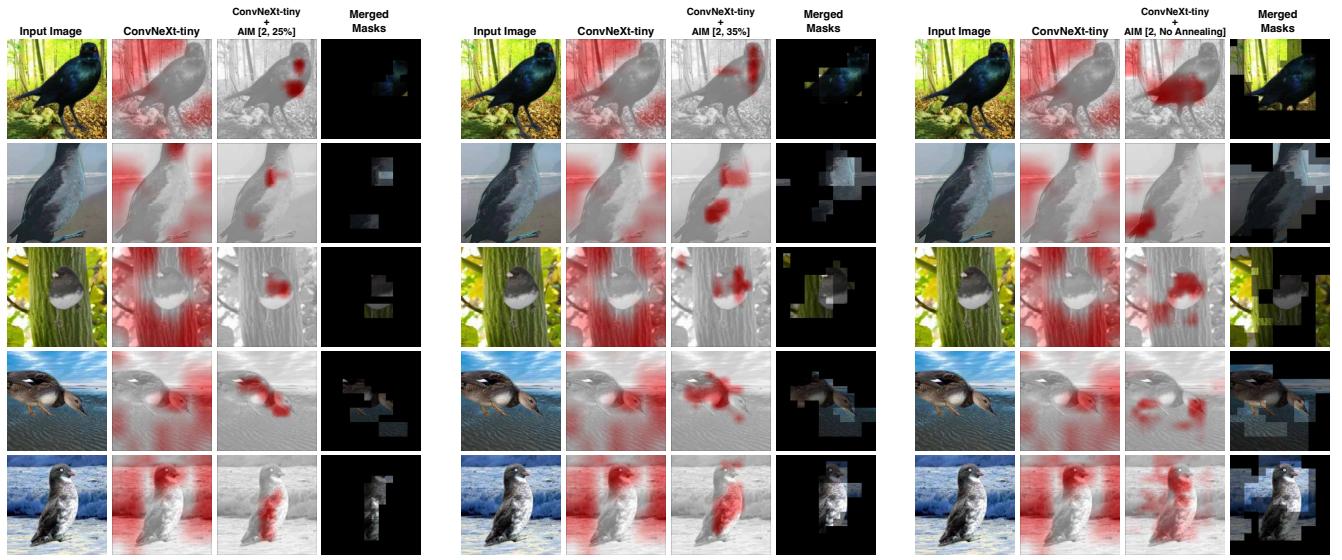


Figure 23. **Qualitative results on the Waterbirds-100% dataset [28].** Comparison of GradCAM attribution maps between ConvNeXt-tiny+AIM models with different mask active-area thresholds, along with the generated merged masks.

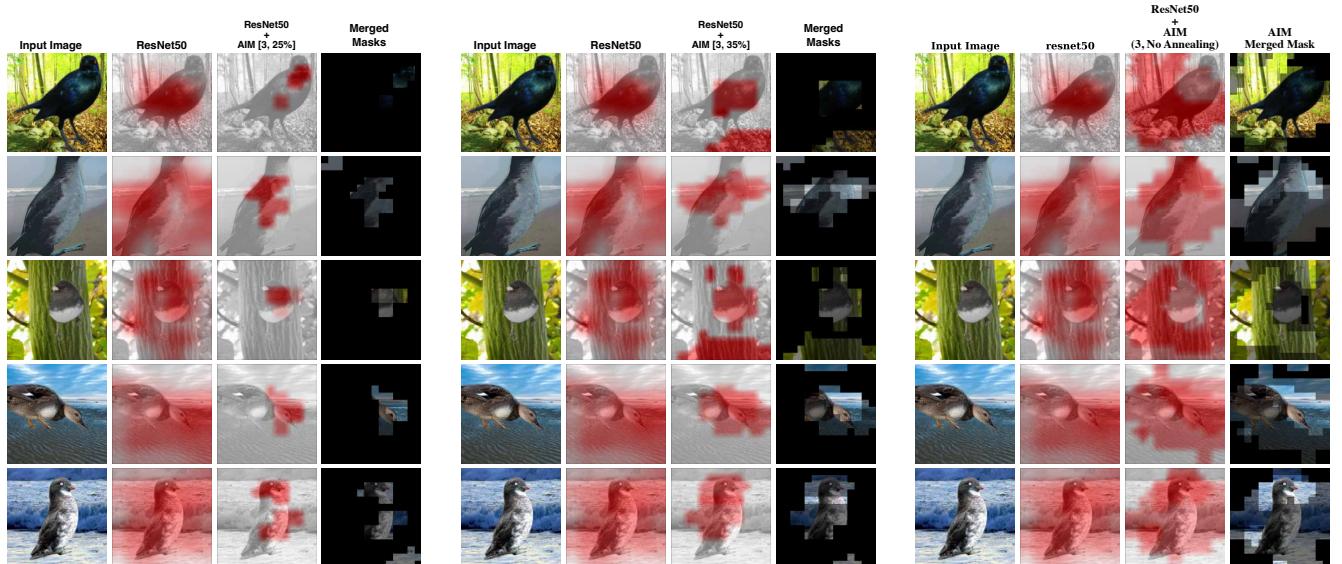


Figure 24. **Qualitative results on the Waterbirds-100% dataset [28].** Comparison of GradCAM attribution maps between ResNet50+AIM models with different mask active-area thresholds, along with the generated merged masks.

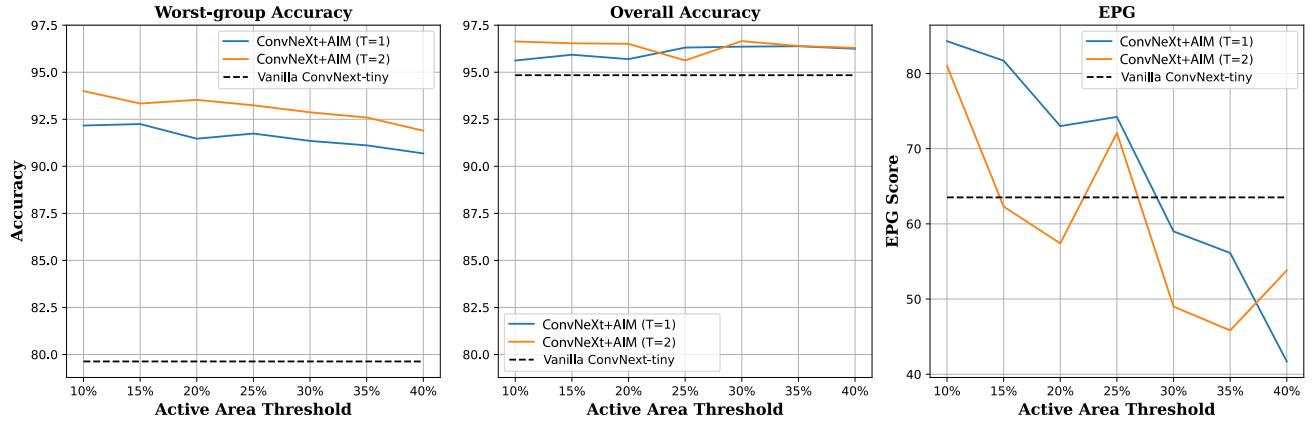


Figure 25. The first figure (on the right) illustrates how the worst-group accuracy of ConvNeXt-tiny+AIM ( $T=1$ ) and ConvNeXt-tiny+AIM ( $T=2$ ) changes with varying active-area thresholds. It shows that as the threshold increases, the worst-group accuracy decreases, while the change in overall accuracy is less pronounced. However, the EPG score also decreases with higher active-area thresholds. Despite this decline, the worst-group accuracy does not experience a significant drop, decreasing by approximately 3%.

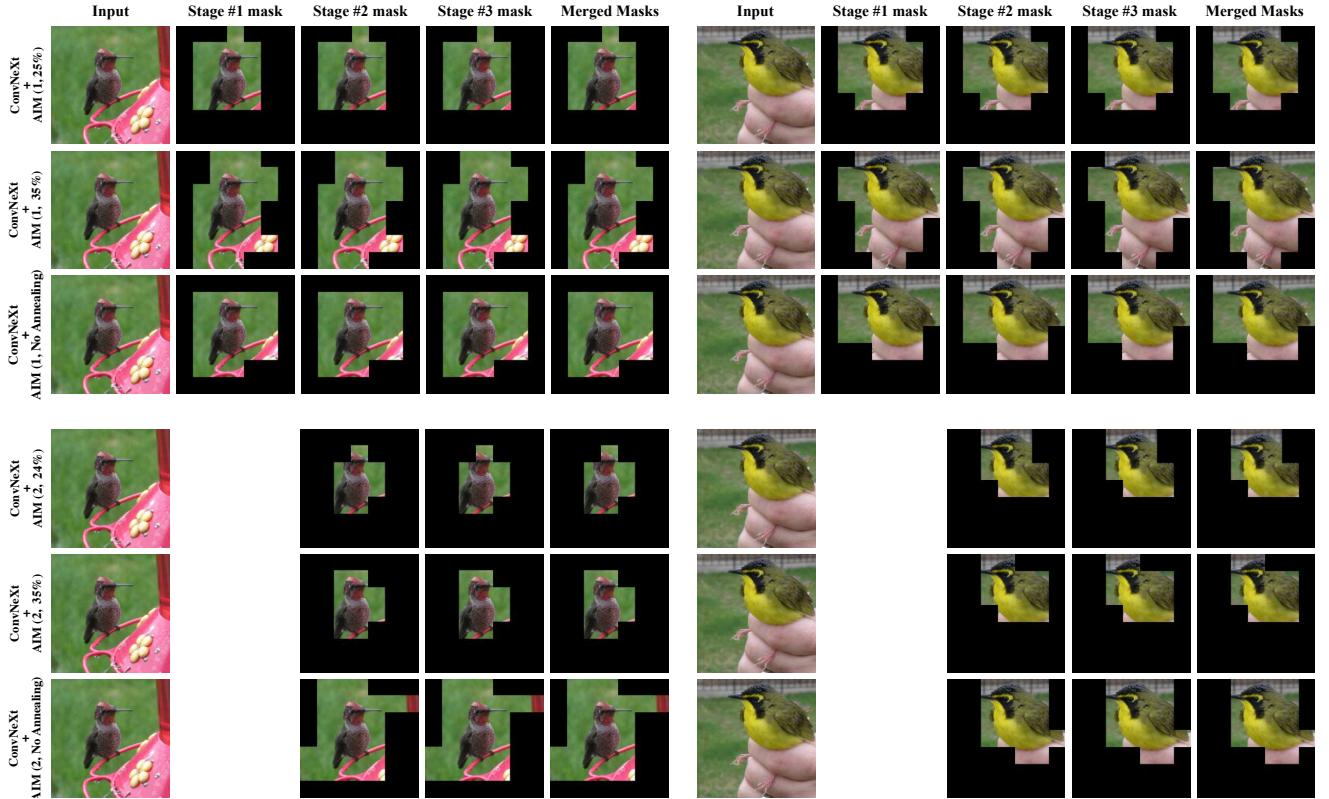


Figure 26. **Qualitative Comparison** between the different architectural configurations of AIM when using masks from the last layer. This figure illustrates the application of masks generated at the last stage (stage #3) and used in the subsequent stages after upsampling.

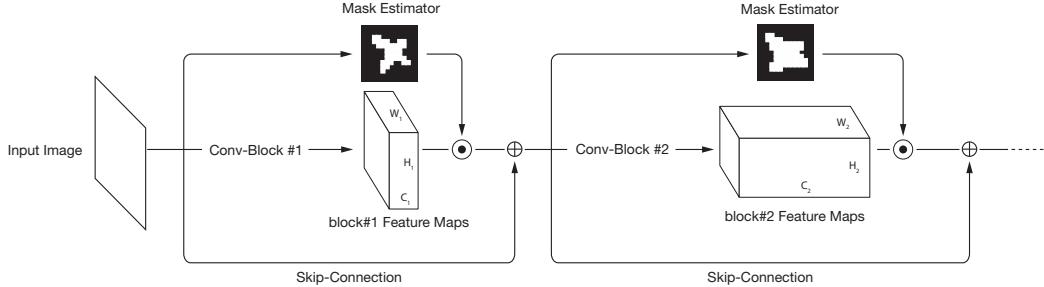


Figure 27. The illustration depicts the architecture of the bottom-up bottlenecking model, adapted from [49], highlighting the flow of data through the model, including skip connections and mask estimators. Each main convolutional block in the network consists of three branches: the first is a mask estimator that employs the Gumbel-softmax trick to predict a binary mask, the second is the original convolutional block, and the third is a skip connection. The generated masks are applied to the output of the convolutional block, resulting in spatially sparse feature maps. Subsequently, the skip connection performs an element-wise summation between these sparse feature maps and the block's input, transforming the sparse maps back into dense ones.

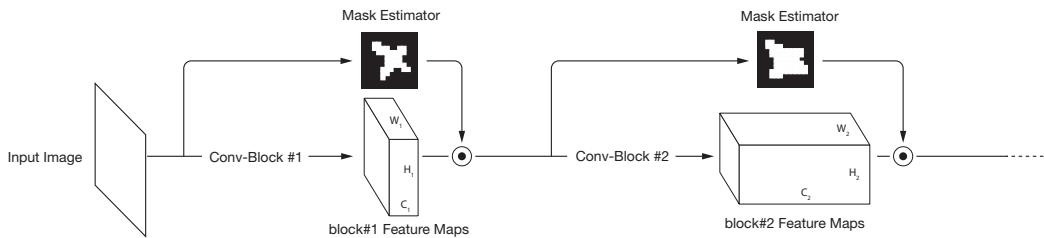


Figure 28. **Bottom-up AIM Network** An illustration showing the same bottom-up bottlenecking model's architecture from Figure 27 but without the skip-connection to maintain the sparsity of the feature maps.



Figure 29. **Bottom-up AIM Network Fails to Generate Focused Masks** This figure illustrates the masks generated at different stages of the ConvNeXt-tiny backbone within the Bottom-up AIM network. Even when we limit the active area region of the masks to 25% and 35% using mask active-area loss annealing, the masks from the earlier blocks remain fully activated, covering the entire image. Only the mask from the last block effectively focuses on the relevant region. This demonstrates that the Bottom-up AIM network does not produce localized, task-related masks in its earlier stages.

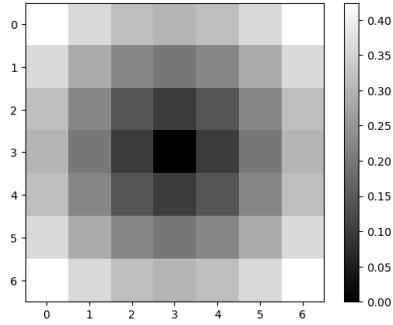


Figure 30. An illustration showing the weight matrix of a convolutional layer with one ( $7 \times 7$ ) filter, initialized using the proposed idea of weighting the convolutional filter weights according to their spatial distance from the center of the filter kernel. Specifically, we assign larger initial values to the weights corresponding to the edges of the filters than to those closer to the center, effectively biasing the filters to be more responsive to features in the outer areas of the input.

Section 1 of 6

## Evaluating AI Visual Focus: A User Perception Study

**Welcome to Our Interpretability Study!**

Thank you for taking the time to participate in this research study. We are exploring how well different Artificial Intelligence (AI) models can "explain" their decisions by showing which parts of an image they focus on.

**What is this study about?**

Our goal is to develop AI models that genuinely focus on the **object of interest** within an image, rather than relying on misleading background details or **spurious correlations/features** when making decisions. Your feedback is crucial in helping us evaluate a new AI approach designed to achieve this.

**What will you be asked to do?**

In this study, you will be presented with a series of images. For each image, you will also see two "attribution maps." An attribution map is a visualization (like a heatmap) overlaid on the image that highlights the regions the AI model considered important. In these maps, "hotter" colors (such as red and orange) indicate areas the AI considered highly important for its decision, while "cooler" colors (like blue and green) represent areas of lower importance. Areas with no color overlay were considered least important or irrelevant by the AI.

- One attribution map will be from our new model.
- The other will be from a standard baseline model.

Your task will be to compare these two attribution maps and select the one that, in your opinion, **more accurately and clearly focuses on the main object in the image**, and less on irrelevant parts.

**How long will it take?**

We estimate this study will take approximately **10 minutes** to complete.

**Your Participation is Valued and Confidential:**

- Participation in this study is completely voluntary.
- All your responses will be kept anonymous and confidential. The data collected will be used solely for research purposes.
- There are no right or wrong answers; we are interested in your genuine perception and opinion.

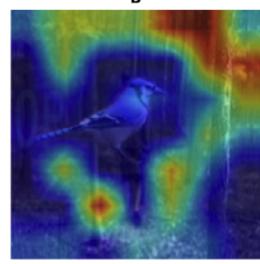
By clicking "Next" and proceeding with the study, you indicate that you have read and understood the information above and consent to participate.

If you have any questions before, during, or after the study, please feel free to contact Eyad Alshami at [ealshami@mpi-inf.mpg.de](mailto:ealshami@mpi-inf.mpg.de)

Thank you again for your valuable contribution!

Figure 31. The user study welcome page, which provided participants with initial instructions.

**In this example, which attribution map (image A or image B) do you think does a better job of \* focusing primarily on the main object and less on other parts of the image?**



A

B

Figure 32. An example trial from our user perception study. Participants were shown the original image (left) and two GradCAM attribution maps from the baseline ConvNeXt (right) and our AIM-equipped model (center). In this case, the baseline model is distracted by the spurious forest background, while our method correctly localizes the waterbird. This clear distinction in focus led users to significantly prefer our model’s explanations.